7th Conference on Algebra and Coalgebra in Computer Science

CALCO 2017, June 14–16, 2017, Ljubljana, Slovenia

Edited by

Filippo Bonchi

Barbara König

Editors

Filippo Bonchi	Barbara König
Dipartimento di Informatica	Fakultät für Ingenieurwissenschaften, Abteilung Informatik und Angewandte Kognitionswissenschaft
Universitá di Pisa	Universität Duisburg-Essen
filippo.bonchi@gmail.com	barbara_koenig@uni-due.de

ACM Classification 1998
F. Theory of Computation, F.1.1 Models of computation, F.3 Logics and Meanings of Programs, F.3.2 Semantics of Programming Languages – Algebraic Approaches to Semantics

ISBN 978-3-95977-033-0

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at http://www.dagstuhl.de/dagpub/978-3-95977-033-0.

Publication date
November, 2017

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Digital Object Identifier: 10.4230/LIPIcs.CALCO.2017.0

ISBN 978-3-95977-033-0 **ISSN 1868-8969** **http://www.dagstuhl.de/lipics**

LIPIcs – Leibniz International Proceedings in Informatics

LIPIcs is a series of high-quality conference proceedings across all fields in informatics. LIPIcs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

ISSN 1868-8969

http://www.dagstuhl.de/lipics

Contents

7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017).
Editors: Filippo Bonchi and Barbara König
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Tools Papers

Preface

This volume contains the proceedings of the 7th Conference on Algebra and Coalgebra in Computer Science, CALCO'17, held in Ljubljana, Slovenia, 14-16 June 2017. Previous CALCO conferences have been held in Swansea (Wales, 2005), Bergen (Norway, 2007), Udine (Italy, 2009), Winchester (UK, 2011), Warsaw (Poland, 2013) and Nijmegen (The Netherlands, 2015).

CALCO is a high-level, bi-annual conference formed by joining the forces and reputations of CMCS (International Workshop on Coalgebraic Methods in Computer Science), and WADT (Workshop on Algebraic Development Techniques). It aims to bring together researchers and practitioners with interests in foundational aspects, and both traditional and emerging uses of algebra and coalgebra in computer science.

This volume contains 22 accepted regular papers, a tools paper and two articles contributed by the invited speakers: Alex Simpson and Nicoletta Sabadini. Thanks to the colocation with MFPS, CALCO also had a joint session on behavioural metrics with a tutorial given by James Worrell and talks by Vincent Danos, Catuscia Palamidessi and Marco Gaboardi. We thank all the speakers for their stimulating talks.

In addition, CALCO has a tradition of Early Ideas talks, allowing the presentation of work in progress and original research proposals. In 2017 CALCO had ten Early Ideas talks. We would like to thank Daniela Petrisan and Till Mossakowski for serving as Early Ideas and Tools chairs. CALCO, Early Ideas and Tools were supported by the EasyChair system that greatly facilitated the CALCO conference submission and program selection process.

We are grateful to the programme committee of CALCO, to the programme committees of the satellite events and the external reviewers for their hard work. The programme committee of CALCO was composed of the following members:

Andrej Bauer
Filippo Bonchi (co-chair)
Marcello Bonsangue
Corina Cîrstea
Robin Cockett
Andrea Corradini
Sergey Goncharov
Helle Hansen
Ichiro Hasuo
Tobias Heindel
Tom Hirschowitz
Bart Jacobs
Bartek Klin
Barbara König (co-chair)
Alexander Kurz
Paul Levy
Radu Mardare
Stefan Milus
Samuel Mimram
Till Mossakowski
Larry Moss
Daniela Petrisan
John Power
Grigore Rosu
Jan Rutten
Peter Selinger
Alexandra Silva
Paweł Sobociński
Ana Sokolova

Finally, we would like to express our thanks to the local organizers, Andrej Bauer and Matija Pretnar, for their efforts in bringing such a wonderful conference to fruition.

Filippo Bonchi and Barbara König

7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017).
Editors: Filippo Bonchi and Barbara König
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

List of Authors

Jiří Adámek

Zeinab Bakhtiari

Henning Basold

Harsh Beohar

Kenta Cho

Corina Cîrstea

Thomas Colcombet

Fredrik Dahlqvist

Clovis Eberhart

Sebastian Enqvist

Brendan Fong

Fabrizio Genovese

Alessandro Gianola

William John Gowers

Brendan Fong

Ichiro Hasuo

Tom Hirschowitz

Helle Hvid Hansen

Bart Jacobs

Tomás Jakl

Achim Jung

Stefano Kasangian

Alexander Knapp

Harald König

Sebastian Küpper

Alexander Kurz

James Laird

Dan Marsden

Stefan Milius

Lawrence Moss

Till Mossakowski

Dirk Pattinson

Daniela Petrisan

Damien Pous

David Reutter

Jurriaan Rot

Nicoletta Sabadini

Lutz Schröder

Fatemeh Seifan

Shunsuke Shimizu

Alex Simpson

Ana Sokolova

David Sprunger

Henning Urbat

Yde Venema

Jamie Vicary

Uwe Wolter

Harald Woracek

Fabio Zanasi

7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017).
Editors: Filippo Bonchi and Barbara König
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Probability Sheaves and the Giry Monad*

Alex Simpson

Faculty of Mathematics and Physics, University of Ljubljana, Slovenia
Alex.Simpson@fmf.uni-lj.si

Abstract

I introduce the notion of *probability sheaf*, which is a mathematical structure capturing the relationship between probabilistic concepts (such as *random variable*) and sample spaces. Various probability-theoretic notions can be (re)formulated in terms of category-theoretic structure on the category of probability sheaves. As a main example, I consider the Giry monad, which, in its original formulation, constructs spaces of probability measures. I show that the Giry monad generalises to the category of probability sheaves, where it turns out to have a simple, purely category-theoretic definition.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases Random variable, conditional independence, category theory, sheaves, Giry monad

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.1

Category Invited Talk

1 Introduction

This article provides technical notes for an invited talk. While these notes present an outline of the main thread of mathematical content of the talk, they do not include the motivation and non-technical discussion that will be given in the talk. Moreover, the talk will assume only basic category theory and probability theory as background, whereas these notes presuppose quite a bit more. The notes also omit the secondary thread of the talk, which concerns parallels between the technical development presented here and one presentation of the *Schanuel topos*, see, e.g., [2], which, in another guise, is well known in computer science as the category of *nominal sets* [3].

2 Sheaves of random variables

Traditionally, a *random variable* is a measurable function from a probability space Ω (the *sample space*) to the measurable space in which the random variable takes its values. In most uses of random variables, however, the sample space plays only an auxiliary role. It serves mainly as a convenient device for manipulating joint probability distributions over all random variables under consideration. The precise identity of the space Ω itself is irrelevant. Indeed, on the contrary, probabilistic notions, such as random variable, enjoy an invariance property under *change of sample space*, which has been argued by Tao to be a characterising

* This research was carried out at the University of Edinburgh, supported by the John Templeton Foundation (grant no. 39465 "Randomness via Information Independence"), and at the University of Ljubljana, supported by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF (Award No. FA9550-14-1-0096) and by the Slovenian Research Agency (research core funding No. P1–0294).

feature of legitimate probabilistic concepts [5]. Tao's principle of invariance under change of sample space is an important guiding principle behind this talk. However, our mathematical formulation will differ from his.

The appropriate notion of change of sample space will be formulated in terms of a category of sample spaces. We avoid mathematically pathological sample spaces, by restricting to *Polish probability spaces* (also known as *standard Borel spaces*).[1]

Recall that a *Polish space* is a topological space that is metrisable as a complete separable metric space. A *Polish probability space* is given by a Polish space Ω together with a probability measure P_Ω on its σ-algebra $\mathcal{B}(\Omega)$ of Borel sets. Henceforth when we say *sample space* we mean Polish probability space. We define the category $\mathbb{P}$ of sample spaces to have:

- *objects:* Polish probability spaces;
- *morphisms from Ω' to Ω:* Borel-measurable functions $q\colon \Omega' \to \Omega$ that are *measure preserving* (i.e., for every Borel $B \subseteq \Omega$, it holds that $P_\Omega(B) = P_{\Omega'}(q^{-1}(B))$).

(In [5], Tao admits arbitrary probability spaces, but restricts maps to *surjective* measure-preserving measurable functions.)

Let A be any Polish space, and Ω any sample space. We define the set $\underline{\mathrm{RV}}(A)(\Omega)$ of A-valued random variables with sample space Ω by:

$$\underline{\mathrm{RV}}(A)(\Omega) \;=\; \{X\colon \Omega \to A \mid X \text{ is Borel measurable}\}/=_{\text{a.e.}}\ ,$$

where $=_{\text{a.e.}}$ is the equivalence relation of almost everywhere equality.

Given an equivalence class $[X] \in \underline{\mathrm{RV}}(A)(\Omega)$ and a map $q\colon \Omega' \to \Omega$, we write $[X].q$ for the equivalence class $[X \circ q] \in \underline{\mathrm{RV}}(A)(\Omega')$ obtained by composition. This is a well-defined operation on equivalence classes. Henceforth, we shall use such explicit equivalence-class notation only where necessary to avoid confusion. In most cases, we shall write X both for the function $X\colon \Omega \to A$, and for its equivalence class.

▶ **Proposition 1.** *The above data defines a functor $\underline{\mathrm{RV}}(A)\colon \mathbb{P}^{\mathsf{op}} \to \mathbf{Set}$; i.e., $\underline{\mathrm{RV}}(A)$ is a presheaf on $\mathbb{P}$.*

We remark that there is an obvious alternative presheaf of random variables, in which the quotienting modulo almost everywhere equality is not performed. Neverthless, the quotiented presheaf seems the more significant of the two. For example, important probabilistic constructions, such as conditional expectation, only define random variables up to almost everywhere equality. Furthermore, the next property, which is fundamental to the ensuing technical development, forces the quotiented presheaf upon us.

▶ **Proposition 2.** *The presheaf $\underline{\mathrm{RV}}(A)$ is **separated** in the sense that, for any $X, X' \in \underline{\mathrm{RV}}(A)(\Omega)$ and map $q\colon \Omega' \to \Omega$, if $X.q = X'.q$ then $X = X'$.*

This follows from the fact that the image of q in Ω has measure 1 in the completion of P_Ω (it is measurable because it is an analytic set).

The theorem below gives an important strengthening of the proposition above. Suppose we have $q\colon \Omega' \to \Omega$ and $Y \in \underline{\mathrm{RV}}(A)(\Omega')$. We say that Y is *q-invariant* if, for every parallel pair $p, p'\colon \Omega'' \to \Omega'$ for which $q \circ p = q \circ p'$, it holds that $Y.p = Y.p'$.

▶ **Theorem 3.** *The presheaf $\underline{\mathrm{RV}}(A)$ is a **sheaf** in the sense that, for every $q\colon \Omega' \to \Omega$ and q-invariant $Y \in \underline{\mathrm{RV}}(A)(\Omega')$, there exists a unique $X \in \underline{\mathrm{RV}}(A)(\Omega)$ such that $Y = X.q$.*

[1] Our development can be adapted to use other classes of space, such as *analytic spaces*, or *standard probability spaces*.

We briefly state the intuitive reading of the material in this section. As in [5], one can view a map $q\colon \Omega' \to \Omega$ as presenting Ω' as an *extension* of the sample space Ω, in which there is additional room for probabilistic variation. Indeed, any $\omega \in \Omega$ in the image of q is expanded by q to the non-empty fibre $q^{-1}(\omega)$ of points in Ω' above it. And since the image of q has measure 1 in Ω, such expansions occur almost everywhere.

Given the above interpretation, Proposition 1 formalises the preservation properties enjoyed by random variables under extension of sample space. This is consistent with the thesis of Tao [5], which may be reformulated as stating that legitimate probabilistic concepts form presheaves. The sheaf property of Theorem 3 also states a property that is desirable, in general, of probabilistic concepts. Suppose we have $q\colon \Omega' \to \Omega$, understood as presenting Ω' as an extension of the sample space Ω. Then the property of $Y \in \underline{\mathrm{RV}}(A)(\Omega')$ being q-invariant asserts that Y does not exploit any of the additional scope for variation available in Ω' beyond that which is already present in Ω. The sheaf property of Theorem 3 then asserts that, in this situation, a (uniquely determined) version of Y is available directly on the sample space Ω itself. Such a restriction from a larger sample space Ω' to a smaller one Ω, in cases in which the additional variation in Ω' is ignored, is a natural property to require of probabilistic concepts in general. Motivated by this, we postulate that probabilistic notions should form *sheaves*, in the general sense introduced in the next section.

3 The category of probability sheaves

Let $F\colon \mathbb{P}^{\mathrm{op}} \to \mathbf{Set}$ be any presheaf. Given any $q\colon \Omega' \to \Omega$ and $y \in F(\Omega')$, we say that y is *q-invariant* if, for every parallel pair $p, p'\colon \Omega'' \to \Omega'$ for which $q \circ p = q \circ p'$, it holds that $y.p = y.p'$. We say that F is a *sheaf* if, for every $q\colon \Omega' \to \Omega$ and q-invariant $y \in F(\Omega')$, there exists a unique $x \in F(\Omega)$ such that $y = x.q$. Because our sheaves are defined over a category of sample spaces, we call them *probability sheaves*.

The category $Sh(\mathbb{P})$ is defined as the full subcategory of $\mathbb{P}$-presheaves on sheaves; i.e., objects are sheaves and morphisms are natural transformations.

In order to understand the structure of $Sh(\mathbb{P})$, we need to look more deeply at the structure of $\mathbb{P}$ itself. Consider a commuting square

$$\begin{array}{ccc} \Omega' & \xrightarrow{r_2} & \Omega_2 \\ {\scriptstyle r_1}\downarrow & & \downarrow{\scriptstyle q_2} \\ \Omega_1 & \xrightarrow[q_1]{} & \Omega \end{array} \tag{1}$$

and write r for the resulting map $r = q_1 \circ r_1 = q_2 \circ r_2$. We say that the commuting square is *independent* if r_1 and r_2 are conditionally independent relative to r, where all maps are considered as random variables over sample space Ω'.

▶ **Theorem 4.** *Every cospan* $\Omega_1 \xrightarrow{q_1} \Omega \xleftarrow{q_2} \Omega_2$ *in* $\mathbb{P}$ *completes to a commuting square*

$$\begin{array}{ccc} \Omega_1 \otimes_\Omega \Omega_2 & \xrightarrow{p_2} & \Omega_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle q_2} \\ \Omega_1 & \xrightarrow[q_1]{} & \Omega \end{array} \tag{2}$$

which enjoys the following characterisation as a universal independent square.

1. *The commuting square (2) is independent; and*
2. *for every independent square (1) completing* $\Omega_1 \xrightarrow{q_1} \Omega \xleftarrow{q_2} \Omega_2$*, there exists a unique map* $\Omega' \xrightarrow{s} \Omega_1 \otimes_\Omega \Omega_2$ *such that* $p_1 \circ s = r_1$ *and* $p_2 \circ s = r_2$*.*

In the proof of the theorem, the Polish space $\Omega_1 \otimes_\Omega \Omega_2$ has the expected set-theoretic pullback

$$\Omega_1 \otimes_\Omega \Omega_2 \;=\; \{(\omega_1, \omega_2) \mid q_1(\omega_1) = q_2(\omega_2)\}$$

as its underlying set, and the probability measure is constructed by integrating fibrewise product measures of the regular conditional probabilities on the fibres of p_1 and p_2 over the conditioning space Ω. For more details, see [4], where this structure is axiomatised as a *local independent product* on $\mathbb{P}$.

Theorem 4 implies *a fortiori* that every cospan in $\mathbb{P}$ completes to a commuting square. This property is known as the *right Ore condition*. It is equivalent to saying that the *dense coverage* on $\mathbb{P}$ is generated by singleton covers, i.e., that the $\mathbb{P}$ carries an *atomic coverage*.[2] Given this, it is immediate from our definition of sheaf, that our sheaves are simply the atomic sheaves over $\mathbb{P}$, and thus $Sh(\mathbb{P})$ is an atomic Grothendieck topos.

The full statement of Theorem 4 strengthens the right Ore condition with a universal property related to conditional independence. Essential use will be made of this strengthening in Section 5 below. The notion of independent square that is associated with this strengthening also permits the following alternative characterisation of the sheaf property.

▶ **Theorem 5.** *The following are equivalent for a presheaf* $F\colon \mathbb{P}^{\mathsf{op}} \to \mathbf{Set}$*.*

1. *F is a sheaf.*
2. *F maps every independent square in $\mathbb{P}$ to a pullback square in* **Set***. (Note that, due to contravariance, for an independent square (1), it is $F(\Omega)$ that is the apex of the resulting pullback square.)*

4 The $\underline{\mathrm{RV}}$ functor

We saw in Section 2 that, for every Polish space A, it holds that $\underline{\mathrm{RV}}(A)$ is a sheaf. In this short section we establish that $\underline{\mathrm{RV}}$ defines a faithful functor from a category of Polish spaces to $Sh(\mathbb{P})$.

A function $f\colon A' \to A$, between two Polish spaces, is said to be *universally measurable* if, for every Borel probability measure $P_{A'}$ on A', and every Borel subset $B \subset A$, it holds that $f^{-1}B$ is measurable in the $P_{A'}$ -completion of $\mathcal{B}(A')$. Every Borel-measurable function is trivially universally measurable, but the converse does not hold. Universally measurable functions are closed under composition. (This is not immediate from the definition.) We write $\mathbf{Pol}_{\mathrm{um}}$ for the category with Polish spaces as objects and universally measurable functions as morphisms.

▶ **Proposition 6.** *The mapping* $A \mapsto \underline{\mathrm{RV}}(A)$ *extends to a faithful functor* $\underline{\mathrm{RV}}\colon \mathbf{Pol}_{um} \to Sh(\mathbb{P})$ *whose action on morphisms defined as follows. For every universally measurable* $f\colon A' \to A$*, sample space* Ω *and* $X \in \underline{\mathrm{RV}}(A)(\Omega)$*, define* $\underline{\mathrm{RV}}(f)(\Omega)(X) = f \circ X$*.*

[2] We adopt the terminology of [2] where, in particular, *coverage* is used as a synonym for *Grothendieck topology*.

The functor $\underline{\mathrm{RV}}$ is far from full. For example, the following morphism from $\underline{\mathrm{RV}}(\mathbf{2})$ to $\underline{\mathrm{RV}}(\mathbf{2})$, where $\mathbf{2} = \{0,1\}$, is not in the image of $\underline{\mathrm{RV}}$.

$$X\colon \Omega \to \{0,1\} \quad \mapsto \quad \omega \in \Omega \ \mapsto \ \begin{cases} 0 & \text{if } \mathbf{P}(X=0) = 1 \text{ or } \mathbf{P}(X=1) = 1 \\ 1 & \text{otherwise } . \end{cases}$$

In general, morphisms from $\underline{\mathrm{RV}}(A)$ to $\underline{\mathrm{RV}}(A')$ can exploit statistical properties of the random variable given as an argument, whereas morphisms in the image of $\underline{\mathrm{RV}}$ have no such capacity.

5 The Giry monad

Giry's classic paper [1] defines two monads of spaces of probability measures. The first is a monad on the category of all measurable spaces. The second is a monad on the category of continuous maps between Polish spaces. The latter construction is defined as follows. It is standard that the set of all Borel probability measures on a Polish space A itself forms a Polish space under the weak topology on probability measures. We write $\mathcal{M}(A)$ for this Polish space of probability measures. In [1], the mapping $A \mapsto \mathcal{M}(A)$ is shown to extend to a monad on the category of continuous maps between Polish spaces. In fact, it can be shown to extend further to a monad on the category $\mathbf{Pol}_{\mathrm{um}}$ of universally measurable maps between Polish spaces. We end this note by showing that it extends beyond this to a monad $\underline{\mathcal{M}}$ on the whole of $Sh(\mathbb{P})$.

Perhaps surprisingly, the monad $\underline{\mathcal{M}}$ can be given a purely category-theoretic definition. For a presheaf $F\colon \mathbb{P}^{\mathsf{op}} \to \mathbf{Set}$ and sample space Ω, define

$$\underline{\mathcal{M}}F\,(\Omega) \ = \ \int^{\Omega'} \mathbb{P}(\Omega',\Omega) \times F(\Omega') \ . \tag{3}$$

Here we use a coend formula for convenience of notation. Nevertheless, since the inside expression is purely contravariant in Ω', the definition simply finds the colimit in $\mathbf{Set}$ of the contravariant functor $\Omega' \mapsto \mathbb{P}(\Omega',\Omega) \times F(\Omega')$.

Because the parameter Ω in (3) is covariant, $\underline{\mathcal{M}}F$ defines a covariant functor from $\mathbb{P}$ to $\mathbf{Set}$. More interestingly, for our purposes, it turns out that $\underline{\mathcal{M}}F$ also defines a contravariant functor; i.e., $\underline{\mathcal{M}}F$ carries the structure of a presheaf. The construction of this presheaf structure exploits *local independent products* in $\mathbb{P}$, as formulated in Theorem 4.

Specifically, suppose we have an equivalence class $[(r,x)] \in \underline{\mathcal{M}}F\,(\Omega)$, where $r\colon \Omega'' \to \Omega$ and $x \in F(\Omega'')$, and a map $q\colon \Omega' \to \Omega$. We need to define $[(r,x)].q \in \underline{\mathcal{M}}F\,(\Omega')$. Consider the local independent product diagram below.

$$\begin{array}{ccc} \Omega' \otimes_\Omega \Omega'' & \xrightarrow{\ p''\ } & \Omega'' \\ {\scriptstyle p'}\downarrow & & \downarrow{\scriptstyle r} \\ \Omega' & \xrightarrow[\ q\ ]{} & \Omega \end{array}$$

Then define

$$[(r,\,x)].q \ = \ [(p',\,x.p'')] \ ,$$

which is indeed well-defined on equivalence classes. Henceforth, we take this contravariant action of $\underline{\mathcal{M}}F$ as basic. Of course it needs to be shown that this action is functorial; i.e., that it defines a presheaf. This property is subsumed under the first item of the theorem below.

▶ **Theorem 7.**
1. *For every presheaf F, it holds that $\underline{\mathcal{M}}F$ is a sheaf.*
2. *The operation $F \mapsto \underline{\mathcal{M}}F$ defines a functor $\underline{\mathcal{M}}$ from $Psh(\mathbb{P})$ to $Sh(\mathbb{P})$.*
3. *The induced endofunctor $\underline{\mathcal{M}}$ on $Sh(\mathbb{P})$ carries the structure of a strong monad.*

Our main result states that $\underline{\mathcal{M}}$ is indeed an extension of $\mathcal{M}$ to the whole of $Sh(\mathbb{P})$.

▶ **Theorem 8.** *There is a natural isomorphism $\underline{\mathrm{RV}}\,\mathcal{M} \cong \underline{\mathcal{M}}\,\underline{\mathrm{RV}}$ which exhibits the functor $\underline{\mathrm{RV}} \colon \mathbf{Pol}_{um} \to Sh(\mathbb{P})$ as strong-monad preserving.*

References

1 M. Giry. A categorical approach to probability theory. In *Categorical Aspects of Topology and Analysis*, pages 68–85. Springer-Verlag, 1982.

2 P. T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Oxford University Press, 2002.

3 A. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*. Cambridge University Press, 2013.

4 A. Simpson. Category-theoretic structure for independence and conditional independence. In *Proceedings of MFPS*, 2017.

5 T. Tao. A review of probability theory. Note0 in 254A – random matrices, 2010. URL: `https://terrytao.wordpress.com/2010/01/01/254a-notes-0-a-review-of-probability-theory/`.

Cospan/Span(Graph): an Algebra for Open, Reconfigurable Automata Networks

Alessandro Gianola[1], Stefano Kasangian[2], and Nicoletta Sabadini[3]

1 **Università degli Studi di Milano, Via Saldini 50, Milano, Italy**
`alessandro.gianola93@gmail.com`
2 **Università degli Studi di Milano, Via Saldini 50, Milano, Italy**
`stefano.kasangian@unimi.it`
3 **Università dell'Insubria, Via Valleggio 11, Como, Italy**
`nicoletta.sabadini@uninsubria.it`

Abstract

Span(**Graph**) was introduced by Katis, Sabadini and Walters as a categorical algebra of automata with interfaces, with main operation being communicating-parallel composition. Additional operations provide also a calculus of connectors or wires among components. A system so described has two aspects: an informal network geometry arising from the algebraic expression, and a space of states and transition given by its evaluation in **Span**(**Graph**). So, **Span**(**Graph**) yields purely compositional, hierarchical descriptions of networks with a fixed topology. The dual algebra **Cospan(Graph)** allows to describe also the sequential behaviour of systems. Both algebras, of spans and of cospans, are symmetrical monoidal categories with commutative separable algebra structures on the objects. Hence, the combined algebra **CospanSpan(Graph)** can be interpreted as a general algebra for reconfigurable/hierarchical networks, generalizing the usual Kleene's algebra for classical automata. We present some examples of systems described in this setting.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases Categories, Automata, Composition, Networks

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.2

Category Invited Talk

1 Introduction

In the last decades, starting with the pioneering approach of C.A. Petri, many formalisms have been proposed in the effort of describing "concurrent" systems, a notion, at least in the opinion of Sabadini and Walters, always very obscure. Some of these formalisms (for example Milner's CCS and in general Process Algebras) insisted correctly on the *compositional* point of view, typical of classical Regular Expressions, by extending the latter with new operators. Unfortunately, the natural correspondence with classical automata, i.e. state/transition systems, was totally lost. This was, in the opinion of Sabadini and Walters, a mistake, since finite automata provide in a natural way the control structure of discrete dynamical (state/transition) systems. Slowly (and - alas - not yet certainly), it has been realized that the semantic object under consideration was *not* a "concurrent" system, but more clearly a distributed network of interacting components/agents/automata. Sabadini-Walters proposed this point of view from the beginning, focusing on the development of an algebra of networks of automata, that should generalize in a natural way the classical algebra of Kleene for

sequential automata. What is missing in Kleene's algebra for automata is the operation of ***communicating parallel***, fundamental to describe any mechanism. Consider for example how two gearwheels interact during their movement by simultaneuos change of their state. We argue that this more fundamental kind of interaction - that it is not input/output message passing, by its nature sequential, but on the contrary a simultaneous change of state - underlies any communication among machines or physical devices. But there is a second issue that should be taken into account in order to provide a compositional description for networks: a network is by default an *open* entity, and it is not possible to build an open system starting with closed ones. Hence, open systems should be taken as basic building blocks, and this forced us to introduce a new notion, *automata with communication ports or interfaces*. But what is the mathematical nature of automata with interfaces? And what is the correct algebra for composing them?

The algebra **Span**(**Graph**) was introduced in [8] as a "parallel" algebra for *automata with interfaces*, I.E. the two fundamental operations are parallel composition with or without communication. The automata are *open* (that is, have interfaces or ports) as is necessary to achieve compositionality. Communication is through synchronization on common actions of the control of different components (not message passing). Communicating automata evolve simultaneously, not in interleaving. In order to have an algebra with at most binary operations the interfaces should be grouped into two sets, what we call respectively the left and right combined ports. There is no implication that left ports are input ports, or that right ports are output ports.

The algebra **Cospan**(**Graph**) was introduced in [9] as a sequential algebra for automata with interfaces, with essentially the Kleene operations of sequential composition, choice and iteration.

In both cases the algebra involved is a well-known one introduced by Jean Benabou in 1967 [3] and developed in 1987 by Carboni, Walters [4] as an algebra of relations, which provides a natural mathematical framework for describing nets of automata, both graphically with circuit-like diagrams, and as terms in a suitable algebra.

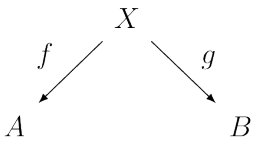
The present paper is essentially a review of previous works of the last twenty years, but it has also some novelties (we suggest that for a better comprehension of this paper and for details, the original papers are available).

The paper is organized as follows: in Section 2, we introduce the algebras of **Span**(**C**) and **Cospan**(**C**), first in the case of an abstract category **C** and then specializing to the case of **Graph**, providing also a series of examples; in Section 3, we discuss closed and open networks relating them to monoidal graphs; in conclusion, in Section 4, we describe open networks with state, showing that it is a reformulation of **Span**(**Graph**).

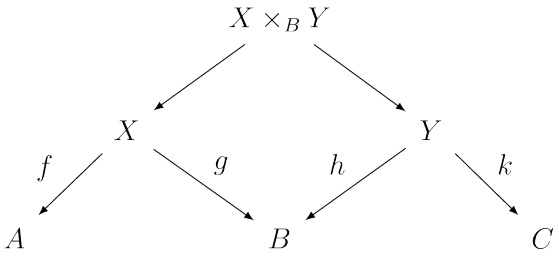
2 Review of the Span and Cospan algebras of automata

2.1 The algebra of spans

▶ **Definition 1.** Given a category **C** with finite limits, we define a new category **Span**(**C**) describing its objects and arrows. Objects of **Span**(**C**) are the same objects of **C**; arrows of **Span**(**C**) from A to B are spans, that is pairs of arrows $(f : X \to A, g : X \to B)$ of **C** with common domain, often written:

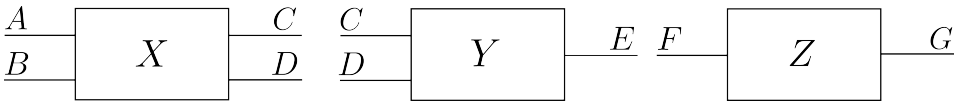


Composition of spans $(f : X \to A, g : X \to B)$ and $(h : Y \to B, k : Y \to C)$ is by pullback (restricted product):

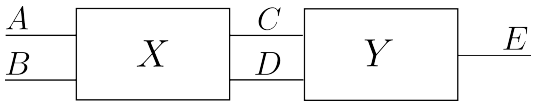


A span $(f : X \to A, g : X \to B)$ will also be written $X : A \to B$, and the composition of span will be indicated with the notation $X;Y : A \longrightarrow C$. The identity span of object A is $(1_A, 1_A)$. The category $\mathbf{Span}(\mathbf{C})$ is actually symmetric monoidal with the tensor of two spans $(f : X \to A, g : X \to B)$ and $(h : Y \to C, k : Y \to D)$ being $(f \times h : X \times Y \to A \times C, g \times k : X \times Y \to B \times D)$.

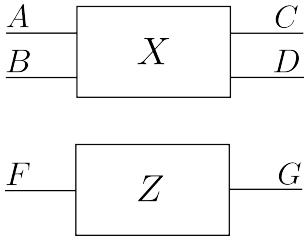
In [7] an informal geometric description (in the style of monoidal category string diagrams) was introduced for the operations in $\mathbf{Span}(\mathbf{C})$. For example, spans $(X \to A \times B, X \to C \times D)$, $(Y \to C \times D, Y \to E)$ and $(Z \to F, X \to G)$ are represented by pictures of *components with ports*:



Then the composite of the first two spans is pictured as:



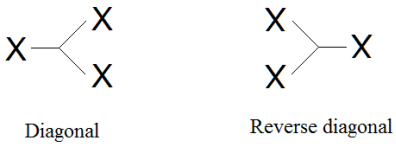
while the tensor of the first and third is pictured as:



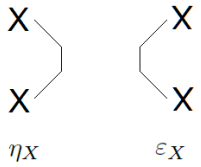
In addition there are constants of the algebra which are pictured as operation on wires which enable the depiction of fanning out of wires and feedback, and hence of general circuit diagrams. We give some examples of constants in $\mathbf{Span}(\mathbf{C})$ which are described also in [7].

- The identity span $1_X : X \longrightarrow X$ has head X and two legs $1_X, 1_X$. It is denoted by a plain wire.

- The span with head X and legs $1_X : X \longrightarrow X$, $\Delta_X : X \longrightarrow X \times X$ is called the diagonal of X and is denoted also $\Delta : X \longrightarrow X \times X$. The span with head X and legs $\Delta_X : X \longrightarrow X \times X$,$1_X : X \longrightarrow X$ is called the reverse diagonal, and is denoted $\nabla : X \times X \longrightarrow X$



- The span with head $X \times Y$ and legs $1_{X\times Y} : X \times Y \longrightarrow X \times Y$, $p_X : X \times Y \longrightarrow X$ is denoted p_X, is called a projection and is pictured by the termination of the wire Y. There is also a similarly defined reverse projection denoted p_X^*.
- Consider the terminal object, denoted I: in case of $\mathbf{C} = \mathbf{Graph}$, the terminal graph has one vertex and one edge, by necessity a loop. The span with head X and legs $! : X \longrightarrow I$, $\Delta_X : X \longrightarrow X \times X$ is called η_X. The span with head X and legs $\Delta_X : X \longrightarrow X \times X$, $! : X \longrightarrow I$ is called ε_X The two spans are pictured in the following figure:



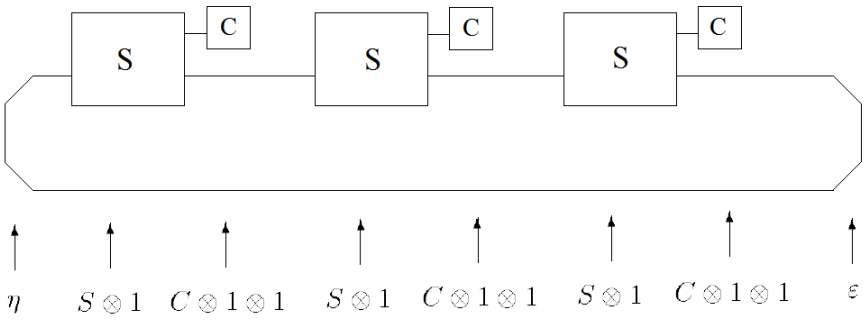
Technically, η and ε are the unit and counit of the self-dual compact-closed structure on $\mathbf{Span}(\mathbf{C})$. They permit a feedback operation on distributed systems.

The correspondence between constants and operations, and the geometric representations given above, result in the fact that expressions in the algebra have corresponding circuit or system diagrams. This is clarified in the following example.

▶ **Example 2.** Given spans $S : X \longrightarrow X \times X$ and $C : X \longrightarrow I$, the expression

$$\eta_X ; (S \otimes 1_X) ; (C \otimes 1_X \otimes 1_X) ; (S \otimes 1_X) ; (C \otimes 1_X \otimes 1_X) ; (S \otimes 1_X) ; (C \otimes 1_X \otimes 1_X) ; \varepsilon_X$$

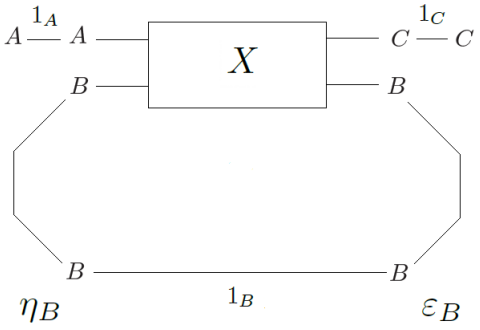
has system diagram (which graphically forms a feedback):



In the following, when we will consider **Span**(**Graph**) as an algebra of automata where parallel composition with or without communication are the main operations, we will use the expression *abstract parallel feedback* referring to the following definition (given by using the constants of **Span**(**C**)):

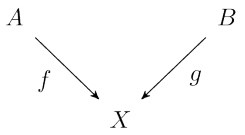
▶ **Definition 3.** Given a span $X : A \times B \longrightarrow C \times B$, we call *abstract parallel feedback* with respect to B, denoted by $AbPfb_B(X)$, the span denoted by the following algebraic expression:

$$(1_A \otimes \eta_B); (X \otimes 1_B); (1_C \otimes \varepsilon_B)$$

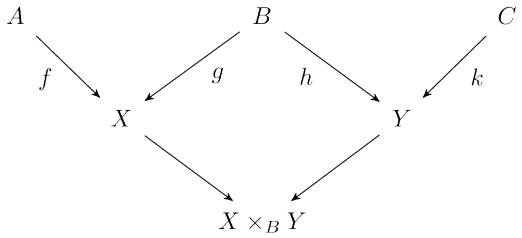


2.2 The algebra of cospans

▶ **Definition 4.** There is a dual construction **Cospan**(**C**) for categories **C** with finite colimits. In fact, **Cospan**(**C**) = **Span**($\mathbf{C}^{\mathbf{op}}$), but it is better seen describing explicitly its objects and arrows. Objects of **Cospan**(**C**) are the same as objects of **C**; arrows of **Cospan**(**C**) from A to B are cospans, that is, pairs of arrows $(f : A \to X, g : B \to X)$ of **C** with common codomain, also written as:

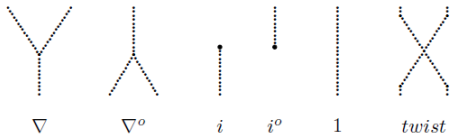


Composition (which is also called *restricted sum*) of $(f : A \to X, g : B \to X)$ and $(h : B \to Y, k : C \to Y)$ is by pushout (glued sum):



The identity cospan of object A is $(1_A, 1_A)$.

Again there are constants of the algebra which are pictured as operation on wires which enable the depiction of joining of wires and sequential feedback. Graphically, we can present these operations as the picture below:



As before, when we will consider **Cospan**(**Graph**) as an algebra of automata where sequential compositions are the main operations, we will use the expression *sequential feedback*.

2.3 Span(Graph), a parallel algebra of automata

Surprisingly, **Span**(**C**), when **C** is the category of (finite) directed graphs (**Span**(**Graph**)), provides a very natural mathematical framework for describing the composition of *automata with interfaces* (or communication ports, as in circuit theory). Consider a span of graphs $(\delta_0 : X \to A, \delta_1 : X \to B)$. The graph X may be considered as the graph of states and transitions of an automata with interfaces, and it is called the *head* of the span. The graph A is the graph of states and transitions of the combined left ports and B is the graph of states and transitions of the combined right ports. The graph morphism δ_0 associates to a state and to a transition of the automaton X the corresponding state and transition of the left ports A; the morphism δ_1 does the same for the right ports.

For all the examples of this paper the left and right ports have only one state so that we tend to ignore that; then δ_0 and δ_1 are a double labelling of the transitions of the automaton X by transitions on the left ports and transitions on the right ports. More intuitively each transition of the component has an effect on all its interfaces, maybe the null effect ε.

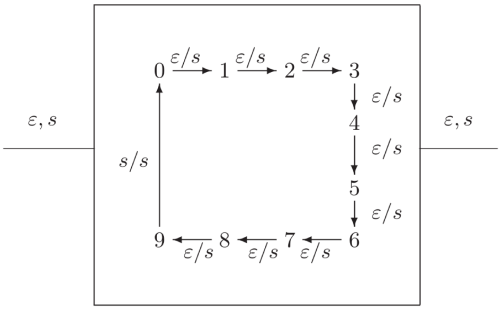
In the case that the left and right ports have one state, the operations of composition and tensor of spans have a simple description in terms of operations on automata. The tensor of two automata has states being pairs of states, one of each automata, and has as transitions pairs of transitions between the corresponding pairs of states. The composition of automata has similarly states being pairs of states, and transitions being pairs of transitions but *only those pairs of transitions whose labels on the connected ports are the same.* In the following, we will call the span composition *parallel composition with communication* and the tensor *parallel composition without communication.*

We give the definition of *parallel feedback* also in the concrete case of **Span**(**Graph**):

▶ **Definition 5.** Given a span of graphs $X : A \times B \longrightarrow C \times B$, we call *parallel feedback* with respect to B, denoted by $Pfb_B(X)$, the span of graphs obtained in the following way: the head of $Pfb_B(X)$ is the graph whose vertices are the vertices of X and whose arcs are the arcs x of X such that $(pr_B \circ \delta_2)(x) = (pr_B \circ \delta_1)(x)$; the interfaces of $Pfb_B(X)$ are the functions $pr_A \circ \delta_1$ and $pr_C \circ \delta_2$.

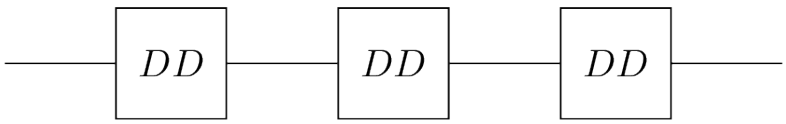
The diagrammatic representation of $Pfb_B(X)$ involves joining the right interface B to the left interface B.

▶ **Example 6** (Decimal Counter)**.** In the following example the left and right ports are both graphs with one state and two transitions ε and s which are displayed on the ports. The automaton has ten states and ten transition (in a circle through the states). The morphisms δ_0 and δ_1 are indicated by doubly-labelling the transitions of the automaton (so, for example, $\delta_0(0 \to 1) = \varepsilon$ and $\delta_1(0 \to 1) = s$).

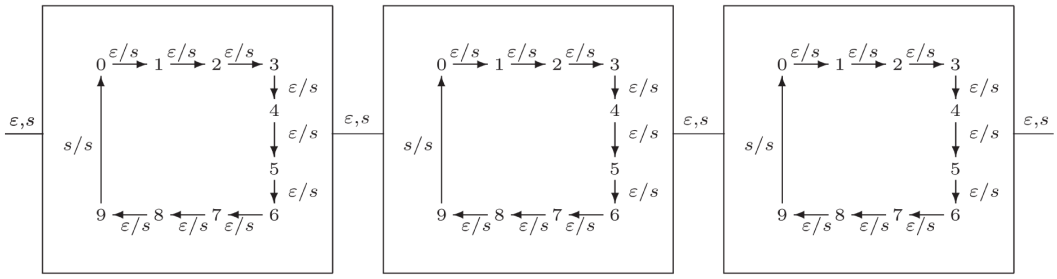


We are actually interested in a variation of this example which we shall call *DecimalDigit* or more shortly *DD* which has the same ports but in addition to the ten transitions above also ten loops one on each state, each labelled $(\varepsilon/\varepsilon)$.

Using composition in **Span**(**Graph**) we can form a new automaton *DecimalCounter* as the span composition of (for simplicity only) three *DD*'s:



or in more detail (though omitting the loops) as:



We notice that a state of *DecimalCounter* is a triple of states one of each DD, and a transition of *DecimalCounter* is a triple of simultaneous transitions with the property that the *labels on the connected ports agree*.

For example, starting in state $(0, 1, 8)$ the following is a sequence of transitions of *DecimalCounter*:

$$(0, 1, 8) \xrightarrow{(\varepsilon/s)} (0, 1, 9) \xrightarrow{(\varepsilon/\varepsilon)} (0, 1, 9) \xrightarrow{(\varepsilon/s)} (0, 2, 0)$$

(remember the ε/ε loops on each state of DD) so a transition s on the right-most port increments the counter. In fact, starting from state $(0, 0, 0)$, after 123 transitions labelled s on the right, the state of the network is $(1, 2, 3)$.

Notice that the transition

$$(9, 9, 9) \xrightarrow{(s/s)} (0, 0, 0)$$

occurs in one step.

▶ Remark. If one wants a parallel algebra of automata which also includes initial and final states of the automata, a slight variation of the above is possible: instead of the category of graphs take the category $(I + J)\backslash\mathbf{Graph}$ whose objects are graph morphisms $(I + J) \to X$, that is consist of two graph morphisms $I \to X$ and $J \to X$, to be thought of as the initial states and final states of X. Then the parallel algebra is $\mathbf{Span}((I + J)\backslash\mathbf{Graph})$.

▶ Remark. In [1], [2] a probabilistic version of $\mathbf{Span}(\mathbf{Graph})$ and $\mathbf{Cospan}(\mathbf{Graph})$ is described, and in that context communicating parallel composition involves conditional probability, arising from the fact that incompatible transition cannot occur and hence probabilities must be normalized. This provides a generalization of Rabin and Segala-Lynch probabilistic automata and this yields a compositional theory of Markov chains.

▶ Remark. In [5] *timed actions* with different duration have been considered. Composition is obtained by considering a linear (w.r.t. transitions) number of extra "internal" states. The intended meaning is that a component that interacts with a "faster" one could be still doing an action (hence being in an internal state) when the other one has completed the transition. We give a simple example:

▶ Example 7 (Two actions which synchronize with an arbitrary duration). Consider two automata $\mathcal{G}_1$, $\mathcal{G}_2$ and two "non-atomic" actions, one of $\mathcal{G}_1$, one of $\mathcal{G}_2$. The action of $\mathcal{G}_1$ is $\{a/b : 0 \to 1, \varepsilon/\varepsilon : 1 \to 1, d/e : 1 \to 2\}$; the action of $\mathcal{G}_2$ is $\{b/c : 0 \to 1, \varepsilon/\varepsilon : 1 \to 1, e/f : 1 \to 2\}$. In the restricted product $\mathcal{G}_1 \cdot \mathcal{G}_2$ these actions synchronize but with arbitrary duration. A typical behaviour is $(0, 0) - a/c \to (1, 1) - \varepsilon/\varepsilon \to (1, 1) - \varepsilon/\varepsilon \to ... - \varepsilon/\varepsilon \to (1, 1) - d/f \to (2, 2)$.

Note. Simple modifications of the algebra allow the description of networks in which the tangling of connectors may be represented, yielding a connection with the theory of knots [11].

2.4 Cospan(Graph), a sequential algebra of automata

Analogously, when $\mathbf{C}$ is the the category of (finite) directed graphs, $\mathbf{Cospan}(\mathbf{Graph})$ provides a sequential calculus for automata that generalizes Kleene's one. Consider a cospan of graphs $(\gamma_0 : E \to X, \gamma_1 : F \to X)$. The graph X may be considered as the graph of states and transitions of an (unlabelled) automaton. The graph E is the graph of *initial* states and transitions and F is the graph of *final* states and transitions. In all the examples considered E and F have only states and not transitions. The graph morphisms γ_0 and γ_1 are often inclusion morphisms of the initial and final states in X.

▶ Remark. If one wants a sequential algebra of automata which have labelled transitions, a slight variation of the above is possible: instead of the category of graphs take the category $\mathbf{Graph}/(A \times B)$ whose objects are graph morphisms $X \to A \times B$, that is consist of two graph morphisms $X \to A$ and $X \to B$, to be thought of as the left and right labellings of X. Then the sequential algebra is $\mathbf{Cospan}(\mathbf{Graph}/(A \times B))$.

2.5 Cospans and spans of graphs

The two algebras we have described may be combined in a natural way. Consider four graph morphisms $(\delta_0 : X \to A, \delta_1 : X \to B, \gamma_0 : E \to X, \gamma_1 : F \to X)$. From these we may obtain an arrow in the parallel algebra $\mathbf{Span}(\mathbf{Graph})\backslash(E + F)$, and also in the sequential algebra $\mathbf{Cospan}(\mathbf{Graph}/(A \times B))$, and hence we may apply both sequential and parallel operations to such automata, obtaining hierarchical nets of automata with evolving geometry. There is a distributive law of parallel composition over sequential, that will be used in the following example. For some details of this see [9].

▶ **Example 8** (Distributed Sort Algorithm)**.** In [9], it has been described in full details an example of reconfigurable network, that is a Distributed Sort Algorithm: an atomic sort A receives a stream of items to be sorted; if the atomic sort gets full, a new network gets activeted in which a divert component D, two atomic sorts A and the merge component M act in parallel. The whole system is the solution of a recursive equation

$$S = A + D \cdot (A \times S) \cdot M$$

which is expanded using the *distributive laws* (between products and the restricted sums) in the following way

$$S = A + D \cdot (A \times S) \cdot M$$

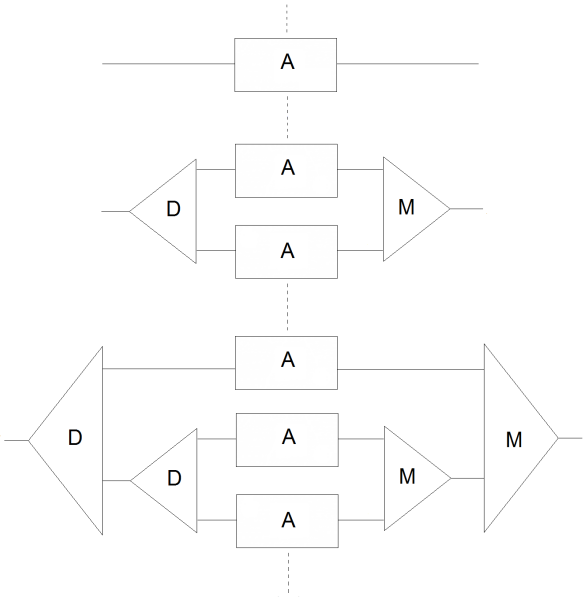
$$= A + D \cdot (A \times (A + D \cdot (A \times S) \cdot M)) \cdot M$$

$$= A + D \cdot (A \times A) \cdot M + D(\cdot(A \times D \cdot (A \times S) \cdot M)) \cdot M$$

$$= .....$$

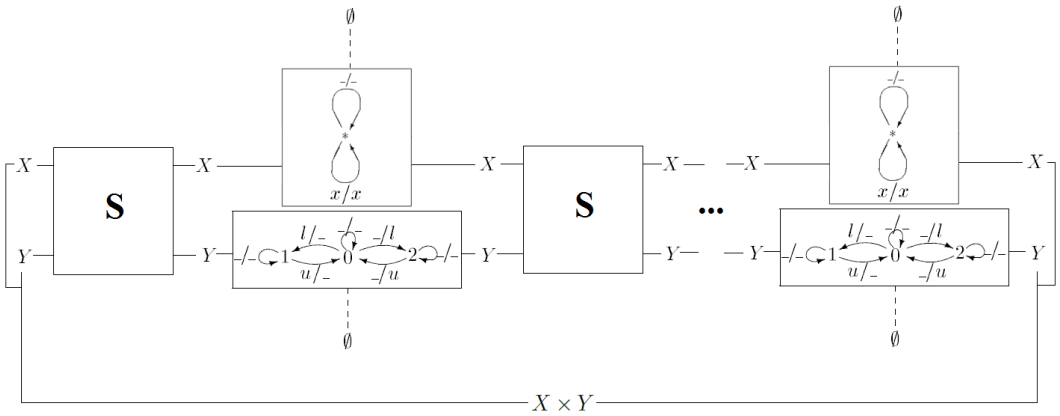
$$= A + D \cdot (A \times A) \cdot M + D(\cdot(A \times D \cdot (A \times A) \cdot M)) \cdot M + ...$$

In this equation the variables are Cospan/Span automata. It gives rise to a network that can be graphically represented as follows:



Now, we are going to see a concrete example of parallel and sequential feedbacks in **Cospan/Span(Graph)**.

▶ **Example 9** (Sofia's Birthday Party)**.** This example [9] is a generalization of the usual Dining Philosophers Problem: instead of a circle of n philosophers around a table with as many forks, we consider a circle of n seats around a table separated by forks. There are $k \leq n$ children: the protocol of each child is the same of a philosopher, but, in addition, if a child is not holding a fork and has an empty seat on the right, he can change seats. In the following picture we give a global representation of the automaton, in which the component S can either be a child (on a seat) or an empty seat:



3 Networks

3.1 Closed networks

In [12] the notion of closed network is formalized by the concept of monoidal graph given in the following definition:

▶ **Definition 10.** A *monoidal graph* $\mathbf{M}$ consists of two finite sets M_0 (wires) and M_1 (components) and two functions $domain : M_1 \to M_0^*$ and $codomain : M_1 \to M_0^*$ where M_0^* is the free monoid on M_0. We call the monoidal graph $\mathbf{M}$ *discrete* when $M_1 = \emptyset$.

▶ Remark. Directed graphs are examples of monoidal graphs in which the domain and codomain functions land in M_0 rather than M_0^*. For these it is usual to speak of elements of M_1 as edges or arcs, and elements of M_0 as vertices. We observe that language of components and wires is more suitable when the domain and codomain functions are words rather than single letters.

An example with $M_1{=}\{A_1,B_1,C_1,D_1,A_2,B_2,C_2,D_2,E\}$ and $M_0{=}\{X_1,Y_1,Z_1,W_1,X_2,Y_2,Z_2,W_2\}$ is:

$A_1 : X_1 \to Y_1$ $\quad$ $B_1 : Y_1 \to Z_1$

$C_1 : Z_1 \to W_1$ $\quad$ $D_1 : W_1 \to X_1$

$A_2 : X_2 \to Y_2$ $\quad$ $B_2 : Y_2 \to Z_2$

$C_2 : Z_2 \to W_2$ $\quad$ $D_2 : W_2 \to X_2$

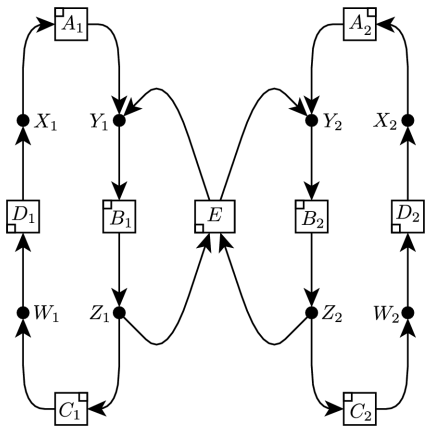
$E : Z_1 Z_2 \to Y_1 Y_2$

▶ **Definition 11.** If $A : X_1 X_2 \cdots X_m \to Y_1 Y_2 \cdots Y_n$ is a component of monoidal graph $\mathbf{M}$, then we call the elements of the set $\{1, 2, \cdots, m\}$ the *left-hand ports* of A, and the elements of $\{1, 2, \cdots, n\}$ the *right-hand ports* of A. Notice that either the set of left-hand ports or of right-hand ports may be empty.

▶ Remark. We insist that, as in **Span**(**Graph**), the left ports and right ports do not have an interpretation as input and output ports respectively. However, in the context of this paper we shall also use the term *input port* for a left-hand port, and *output port* for a right-hand port. For this reason we also indicate input ports by arrows on wires entering the component, and output ports by arrows on wires exiting the component.

The notion of ports of a component gives rise to a different *geometric* representation of monoidal graph which explains why we regard such a graph as a closed network. The elements of M_1 are represented as components with ports, the elements of M_0 are represented as connectors or wires, and the domain and codomain functions describe how the ports are joined to the connectors.

The order on the ports is indicated by a small square in the component nearest to the first left-hand port (in the example below, see in particular the component $E : Z_1 Z_2 \to Y_1 Y_2$). The right-hand ports are taken in the same order as the left-hand ports.



Note. After this section we shall usually omit the indicator of the order of the ports, where the intended order is clear.

▶ Remark. It might be objected that the simple monoidal graph with two wires X, Y and one component $A : X \to Y$ should not be considered as a *closed* network. However, while the component A has left and right ports, the monoidal graph itself does not have specified left and right ports. We will see when we define open networks in the next section that this graph may have external ports specified in many different ways.

We will need later the definition of morphism of monoidal graph.

▶ **Definition 12.** A morphism α from monoidal graph $\mathbf{M}$ to $\mathbf{N}$ consists of two functions $\alpha_1 : M_1 \to N_1$ and $\alpha_0 : M_0 \to N_0$ such that $domain_N \circ \alpha_1 = (\alpha_0)^* \circ domain_M$ and $codomain_N \circ \alpha_1 = (\alpha_0)^* \circ codomain_M$, where $(\alpha_0)^*$ is the monoid homomorphism between free monoids induced by the function α_0.

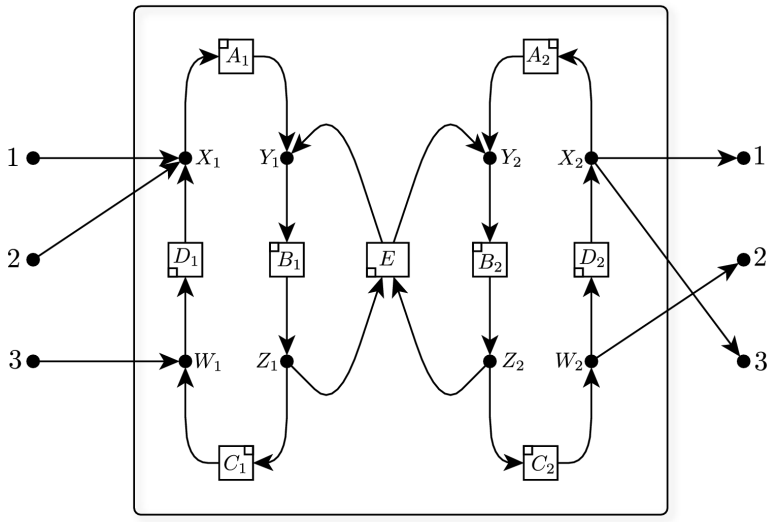
▶ Remark. It is not difficult to verify that monoidal graphs with these morphisms form a category, denoted **MonGraph**, which is in fact a presheaf topos.

3.2 Open networks

In order to show that the globally described notion of network may be also described compositionally, we need a notion of *open* network, which is introduced in [12]. An open network is a monoidal graph together with left and right interfaces. The formal definition is as follows:

▶ **Definition 13.** An open network consists of a monoidal graph $\mathbf{M}$, two sets $S = \{1, 2, \cdots, m\}$ and $T = \{1, 2, \cdots, n\}$ and two functions $\gamma_0 : S \to M_0$, $\gamma_1 : T \to M_0$. We denote the open network as $\mathbf{M} : S \to T$. If both sets S and T are empty, the only content of the network is the monoidal graph, that is, the network is closed.

We sketch an example in which the sets $S = T = \{1, 2, 3\}$. Notice it has the same form as a single component.



An open network has a simple standard categorical description. It is a *cospan of monoidal graphs* between discrete monoidal graphs.

3.3 The algebra of open networks

The algebraic structure that cospans (open networks) admit can be described in the following way: they form the arrows of *a symmetric monoidal category in which every object has a commutative separable algebra structure compatible with the tensor* [4], [8] - what is called Well-Supported-Compact-Closed categories. We first give a definition of WSCC category assuming knowledge of symmetric monoidal categories.

▶ **Definition 14.** A commutative separable algebra in a symmetric monoidal category consists of an object X and four arrows $\nabla : X \otimes X \to X$, $\Delta : X \to X \otimes X$, $n : I \to X$ and $e : X \to I$ making (X, ∇, n) a commutative monoid, (X, Δ, e) a cocommutative comonoid and satisfying the equations

$$(1_X \otimes \nabla)(\Delta \otimes 1_X) = \Delta\nabla = (\nabla \otimes 1_X)(1_X \otimes \Delta) : X \otimes X \to X \otimes X$$
$$\nabla\Delta = 1_X.$$

There are two important derived operations of a commutative separable algebra X, namely $\epsilon : X \otimes X \to I = e\nabla$ and $\eta : I \to X \otimes X = \Delta n$.

▶ Proposition 1. Given an object X with a commutative separable algebra structure the arrows $\eta : I \to X \otimes X$ and $\epsilon : X \otimes X \to I$ satisfy the equations (where τ is the twist of the symmetry):

(i) $(\epsilon \otimes 1_X)(1_X \otimes \eta) = 1_X = (1_X \otimes \epsilon)(\eta \otimes 1_X)$

(ii) $\epsilon\tau = \epsilon$ and $\tau\eta = \eta$.

Proof. We prove only (i), as done in [12]. To see that $(\epsilon \otimes 1_X)(1_X \otimes \eta) = 1_X$ notice that

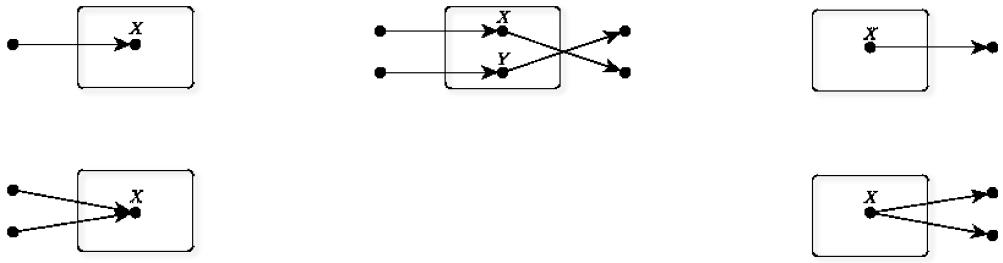
$$(e \otimes 1_X)(\nabla \otimes 1_X)(1_X \otimes \Delta)(1_X \otimes n) = (e \otimes 1_X)\Delta\nabla(1_X \otimes n) = 1_X \cdot 1_X = 1_X;$$

The first equality is an application of the first displayed axiom of definition 2.5 in [8]. The second equality comes from the monoid and comonoid axioms. ◀

▶ Remark. Axiom (i) says that X is a *self-dual object*. The reader can translate these into equations between *string* diagrams, which are representations of expressions. An example is the string diagram for (i):

▶ **Definition 15** (The operations of the algebra of open networks)**.** The open network composition of $\mathbf{M} : S \to T$ with $\mathbf{N} : T \to R$ is composition of cospans of monoidal graphs; that is, the components of the composite are the disjoint union of the components of $\mathbf{M}$ and $\mathbf{N}$ whereas the wires of the composite are the quotient of the disjoint union of the wires of $\mathbf{M}$ and $\mathbf{N}$ obtained by equating the images of elements of T (under γ_1) in $\mathbf{M}$ with the corresponding images of the elements of T (under γ_0) in $\mathbf{N}$. The open network tensor of $\mathbf{M}$ and $\mathbf{N}$ is formed by taking disjoint unions of both wires and components.

Similarly, the constants of the algebra of networks can be given by using the constants of the algebra of the monoidal graph. We don't give a precise definitions but we present them by picturing.

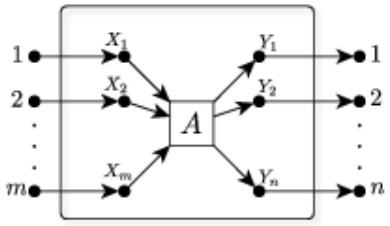


3.4 The algebra of open networks is free

▶ **Theorem 16.** *Given a monoidal graph* **M***, the free symmetric strict monoidal category constructed from* **M** *adjoining commutative separable algebra structures to the objects of* **M** *is the full subcategory of* **Cospan**(**MonGraph**/**M**) *whose objects are discrete monoidal graphs over (labelled in)* **M***.*

Proof. In [10] the special case for graphs rather than monoidal graphs is proved by demonstrating (in Proposition 3.3 of that paper) a normal form for arrows in the free category (with structure). The proof of the theorem that we are describing here (even if it is more general of the one in [10]) requires just a straightforward modification of the normal form in which a sum of edges is replaced by a sum of components of **M**. ◀

We denote this full subcategory as **Csp**(**MonGraph**/**M**). The components of **M** lie in **Csp**(**MonGraph**/**M**) as cospans as follows: if $A : X_1X_2\cdots X_m \to Y_1Y_2\cdots Y_n$ is a component of **M**, then the corresponding cospan is



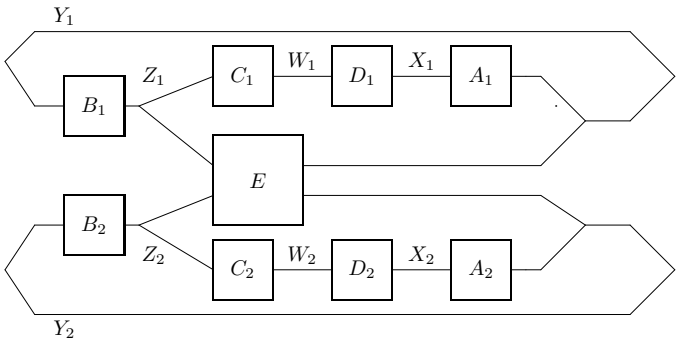
where the elements A, $X_1, X_2, \cdots, X_m$, $Y_1, Y_2, \cdots, Y_n$ are now *labels* - that is, if in the original monoidal graph $X_1 = X_2$ then in the cospan the corresponding wires are distinct but have the same label $X_1 = X_2$ in **M**.

▶ Remark. The first practical use of this theorem is that from the components of **M** we can generate any open (or closed) monoidal graph containing these components by evaluating an expression in the algebra **Csp**(**MonGraph**/**M**) starting from single components.

As an example we describe the monoidal graph **M** of section 3.1 as an expression in **Csp**(**MonGraph**/**M**) in terms of single components. The following expression evaluates to **M**:

$$(\epsilon_{Y_1}\otimes\epsilon_{Y_2})(1_{Y_1}\otimes((\nabla_{Y_1}\otimes\nabla_{Y_2})(A_1D_1C_1\otimes E\otimes A_2D_2C_2)(\Delta_{Z_1}B_1\otimes\Delta_{Z_2}B_2))\otimes 1_{Y_2})(\eta_{Y_1}\otimes\eta_{Y_2})$$

as can be seen by examining the string diagram for this expression:



▶ Remark. A theorem similar to Theorem 1, for ordinary graphs only, was described by Gadducci, Heckel and Llabres [6] for application to graph rewriting.

4 Networks with state

The aim of this section is to attach automata to each of the components and wires of a monoidal graph; that is, to add states and state transitions to each part of the network, as done in [12].

4.1 Adding state to networks

A cospan of monoidal graphs with these extra labelling automata is what we use to model *open networks with state*.

Again this has a standard categorical description. Consider the monoidal category **Span**(**Graph**): composition is pullback, tensor is product. **Span**(**Graph**) is a WSCC category, but now $\Delta : X \to X \times X$ is the span with left-leg the identity and right-leg the diagonal of the product, and $\nabla : X \times X \to X$ is the reverse of the span Δ, as defined in Section 2.1. Let $|\mathbf{Span}(\mathbf{Graph})|$ be the (large) monoidal graph whose wires are the objects of **Span**(**Graph**) (that is, graphs) and whose components are spans of graph morphisms between products of graphs. Then we have the following definition:

▶ **Definition 17.** An open network with state is an arrow in the monoidal category

$$\mathbf{Csp}(\mathbf{MonGraph}/|\mathbf{Span}(\mathbf{Graph})|).$$

Hence, a closed network with state is a morphism of monoidal graphs from **MonGraph** to $|\mathbf{Span}(\mathbf{Graph})|$.

Note that $\mathbf{Csp}(\mathbf{MonGraph}/|\mathbf{Span}(\mathbf{Graph})|)$ is also a WSCC category.

The definition simply means that associated with each wire of an open network there is a graph, and to each component $A : X_1X_2 \cdots X_m \to Y_1Y_2 \cdots Y_n$ a span of graphs between the products of the graphs labelling the wires.

▶ **Definition 18.** The process of forming the global state space of a network with state is the functor preserving the WSCC structure

$$\mathit{globalstate} : \mathbf{Csp}(\mathbf{MonGraph}/|\mathbf{Span}(\mathbf{Graph})|) \longrightarrow \mathbf{Span}(\mathbf{Graph}),$$

which is induced, using the freeness of the domain category, from the inclusion of the components of $|\mathbf{Span}(\mathbf{Graph})|$ in **Span**(**Graph**).

▶ **Theorem 19.** *The functor globalstate can be obtained by calculating a global* limit *in* **Graph**.

A sketch of the proof of the previous Theorem can be found in [12].

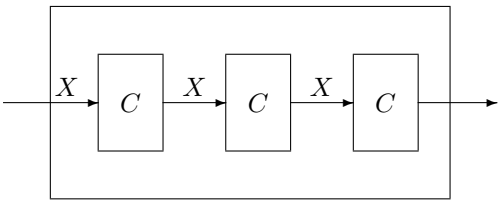
▶ **Corollary 20.** *The systems described by networks with state are the same as* **Span**(**Graph**) *systems.*

Proof. Consider the functor

$$globalstate : \mathbf{Csp}(\mathbf{MonGraph}/|\mathbf{Span}(\mathbf{Graph})|) \longrightarrow \mathbf{Span}(\mathbf{Graph}).$$

An arrow in the domain may be thought of as either (i) algebraically, *an expression in* **Span**(**Graph**) *(the compositional view of a system)* or (ii) geometrically, *a network (cospan of monoidal graphs) labelled in graphs.* The functor *globalstate* may be obtained by calculating either (i) by evaluating the expression in **Span**(**Graph**) or (ii) by calculating a global limit. ◀

▶ **Example 21.** We describe *DecimalCounter*, the decimal counter with three digits given above, in this context. Consider the monoidal graph $\mathbf{M}$ with one component C and one wire X with $C : X \to X$. Then, arrows in $\mathbf{Csp}(\mathbf{MonGraph}/\mathbf{M})$ are open monoidal graphs built out of the open monoidal graph corresponding to this component. The example we wish to consider is the composite $CCC : X \to X$ in $\mathbf{Csp}(\mathbf{MonGraph}/\mathbf{M})$ which is an open monoidal graph with the following graphical representation



Now, by the freeness of $\mathbf{Csp}(\mathbf{MonGraph}/\mathbf{M})$, a monoidal graph morphism from $\mathbf{M}$ to $|\mathbf{Span}(\mathbf{Graph})|$ induces a structure preserving functor

$$\mathbf{Csp}(\mathbf{MonGraph}/\mathbf{M}) \to \mathbf{Csp}(\mathbf{MonGraph}/|\mathbf{Span}(\mathbf{Graph})|)$$

which takes the open monoidal graph $CCC : X \to X$ to a network with state. To describe the decimal counter we choose the following morphism of monoidal graphs $\mathbf{M} \to |\mathbf{Span}(\mathbf{Graph})|$: X goes to the graph with one vertex and edges ε, s and C goes to the span of graphs DD.

The decimal counter now consists of three such automata joined in parallel according to the pattern of the open monoidal graph $CCC : X \to X$.

Acknowledgements. To Bob.

References

1. L. d. F. Albasini, N. Sabadini, and R. F. C. Walters. The compositional construction of Markov processes. *ArXiv e-prints*, January 2009. `arXiv:0901.2434`.
2. L. d. F. Albasini, N. Sabadini, and R. F. C. Walters. The compositional construction of Markov processes II. *ArXiv e-prints*, May 2010. `arXiv:1005.0949`.
3. J. Bènabou. Introduction to Bicategories. *Reports of the Midwest Category Seminar*, 47:1–77, 1967. `doi:10.1007/BFb0074299`.
4. A. Carboni and R.F.C. Walters. Cartesian bicategories I. *Journal of Pure and Applied Algebra*, 49(1):11 – 32, 1987.
5. A. Cherubini, N. Sabadini, and R. F. C. Walters. Timing in the cospan-span model. *Electr. Notes Theor. Comput. Sci.*, 104:81–97, 2004.
6. F. Gadducci, R. Heckel, and M. Llabrés. A bi-categorical axiomatisation of concurrent graph rewriting. *Electronic Notes in Theoretical Computer Science*, 29:80 – 100, 1999. CTCS '99, Conference on Category Theory and Computer Science.
7. P. Katis, N. Sabadini, and R. F. C. Walters. Span(graph): A categorial algebra of transition systems. In *Proceedings of AMAST '97*, pages 307–321, 1997.
8. P. Katis, N. Sabadini, and R.F.C. Walters. Bicategories of processes. *Journal of Pure and Applied Algebra*, 115:141–178, 1997.
9. P. Katis, N. Sabadini, and R.F.C. Walters. A formalization of the IWIM model. In *Proceedings of Coordination Languages and Models 2000*, volume 1906 of *LNCS*, pages 267–283. Springer, 2000.
10. R. Rosebrugh, N. Sabadini, and R.F.C. Walters. Generic commutative separable algebras and cospans of graphs. *Theory and Applications of Categories*, 15(6):164–177, 2005.
11. R. Rosebrugh, N. Sabadini, and R.F.C. Walters. Tangled circuits. *Theory and Applications of Categories*, 26(27):743–767, 2012.
12. N. Sabadini, F. Schiavio, and R.F.C. Walters. On the geometry and algebra of networks with state. *Theor. Comput. Sci.*, 64:144–163, 2017.

On Corecursive Algebras for Functors Preserving Coproducts*

Jiří Adámek[1] and Stefan Milius[†,2]

1 **Institut für Theoretische Informatik**
Technische Universität Braunschweig, Germany
2 **Lehrstuhl für Theoretische Informatik**
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Abstract

For an endofunctor H on a hyper-extensive category preserving countable coproducts we describe the free corecursive algebra on Y as the coproduct of the terminal coalgebra for H and the free H-algebra on Y. As a consequence, we derive that H is a *cia functor*, i.e., its corecursive algebras are precisely the cias (completely iterative algebras). Also all functors $H(-)+Y$ are then cia functors. For finitary set functors we prove that, conversely, if H is a cia functor, then it has the form $H = W \times (-) + Y$ for some sets W and Y.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases terminal coalgebra, free algebra, corecursive algebra, hyper-extensive category

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.3

1 Introduction

Iteration and (co)recursion are of central importance in computer science. A formalism for iteration was proposed by Elgot [11] as iterative algebraic theories. Later Nelson [14] and Tiuryn [15] introduced iterative algebras for finitary signatures which yield an easier approach to iterative theories. For endofunctors H there are two related notions of algebras. *Corecursive algebras* introduced by Capretta et al. [9] are those algebras A such that every recursive equation expressed as a coalgebra for H has a unique solution (i.e., a coalgebra-to-algebra morphism into A). The other notion, *completely iterative algebras* (or *cia*, for short), introduced by the second author [13], are H-algebras A with the stronger property that every recursive equation with parameters in A has a unique solution (Definition 7). Corecursive algebras often fail to be cias. In the present paper we study endofunctors such that every corecursive algebra is a cia – we call them *cia functors*.

Our first result is that every endofunctor preserving countable coproducts and having a terminal coalgebra is a cia functor (Corollary 21). This is based on a description of the free cia on an object Y as a coproduct

$$\nu H + FY$$

of the terminal coalgebra and the free algebra on Y (Theorem 14). We deduce that, for H preserving countable coproducts and having a terminal coalgebra, we obtain cia functors

* A full version of the paper is available at `https://arxiv.org/abs/1703.07574`, [5].
† Supported by Deutsche Forschungsgemeinschaft (DFG) under project MI 717/5-1

$H(-)+Y$ for all objects Y (Corollary 24). All this holds in every *hyper-extensive* base category (Definition 1), e.g., in sets, posets, graphs and all presheaf categories.

In particular, if the base category is also cartesian closed, then $X \mapsto W \times X + Y$ is a cia functor for every pair of objects W and Y. For finitary set functors we prove a surprising converse: the only cia functors are those of the above form $X \mapsto W \times X + Y$.

2 Preliminaries

Throughout the paper H denotes an endofunctor on a hyper-extensive category (recalled below) having a terminal coalgebra

$$t : \nu H \to H(\nu H).$$

By the famous Lambek Lemma [12], the coalgebra structure t is invertible and its inverse makes νH an H-algebra.

We denote by $\mathsf{Alg}\, H$ the category of H-algebras and their morphisms.

▶ **Definition 1** ([2])**.** A category is called *hyper-extensive* if it has countable coproducts which are

(1) *universal*, i.e., preserved by pullbacks along any morphism,

(2) *disjoint*, i.e., coproduct injections are monomorphic and have pairwise intersection 0 (the initial object), and

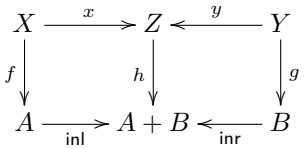
(3) *coherent*, i.e., given pairwise disjoint morphisms $a_n : A_n \to A$, $n \in \mathbb{N}$, each of which is a coproduct injection, then their copairing $[a_n]_{n\in\mathbb{N}} : \coprod_{n\in\mathbb{N}} A_n \to A$ is also a coproduct injection.

▶ **Example 2.** The categories of sets, posets, graphs, and presheaf categories are hyper-extensive.

▶ Remark 3.

(1) We write $A+B$ for the coproduct of the objects A and B and denote coproduct injections by $\mathsf{inl} : A \to A+B$ and $\mathsf{inr} : B \to A+B$.

(2) Recall that a category with finite coproducts is *extensive* if it has pullbacks along coproduct injections and conditions (1) and (2) are satisfied [10]. Equivalently, in a diagram of the following form



the top row is a coproduct if and only if the squares are pullbacks. Another, more compact, equivalent characterization of extensivity states that the canonical functor $\mathcal{C}/A \times \mathcal{C}/B \to \mathcal{C}/(A+B)$ is an equivalence of categories for any pair of objects A and B.

(3) The somewhat technical condition (3) in Definition 1 is not a consequence of the other two. In fact, let $\mathcal{C}$ be the category of Jónsson-Tarski algebras, i.e., binary algebras A whose operation $A \times A \to A$ is a bijection. Then $\mathcal{C}$ has disjoint and universal countable (in fact, all) coproducts but is not hyperextensive [2].

▶ **Definition 4** ([9]). An algebra $a : HA \to A$ is called *corecursive* if for every coalgebra $e : X \to HX$ there exists a unique algebra-to-coalgebra morphism $e^\dagger : X \to A$:

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle a} \\ HX & \xrightarrow[He^\dagger]{} & HA \end{array} \tag{1}$$

▶ **Examples 5.**

(1) The terminal coalgebra νH (considered as an algebra) is obviously corecursive. This is the initial corecursive algebra [9].
Furthermore, let Y be an object of $\mathcal{C}$ and assume that the functor $H(-) + Y$ has a terminal coalgebra TY. Then its structure

$$TY \xrightarrow{\alpha_Y} HTY + Y$$

has an inverse which is the copairing of two morphisms denoted by

$$HTY \xrightarrow{\tau_Y} TY \xleftarrow{\eta_Y} Y.$$

It follows that TY is a coproduct of HTY and Y with the above coproduct injections. It is easy to show that (TY, τ_Y) is a corecursive algebra.

(2) The trivial terminal algebra $H1 \to 1$ is corecursive, and if (A, a) is a corecursive algebra so is (HA, Ha) [9, Prop. 21]. Furthermore, if $\mathcal{C}$ has limits then corecursive algebras are closed under limits in the category of algebras for H [3, Prop. 2.4]. It follows that all members of the *terminal-coalgebra* chain

$$1 \longleftarrow H1 \longleftarrow HH1 \longleftarrow \cdots$$

are corecursive algebras.

(3) A particular instance of point (1) is given by a signature $\Sigma = (\Sigma_n)_{n<\omega}$ of operation symbols with prescribed arity and considering the corresponding polynomial endofunctor H_Σ on Set defined by

$$H_\Sigma X = \coprod_{n<\omega} \Sigma_n \times X^n.$$

For an operation symbol $\sigma \in \Sigma_n$ we write $\sigma(x_1, \ldots, x_n)$ in lieu of $(\sigma, (x_1, \ldots, x_n))$ for elements in the summand of $H_\Sigma X$ corresponding to $n < \omega$. The terminal coalgebra νH_Σ is carried by the set of all Σ-*trees*, i.e., rooted and ordered trees with nodes labeled in Σ such that every node with n children is labeled by an n-ary operation symbol. The algebraic operation of νH_Σ is *tree-tupling*: t^{-1} assigns to $\sigma(t_1, \ldots, t_n)$ with $\sigma \in \Sigma_n$ and $t_i \in \nu H_\Sigma$, $i = 1 \ldots, n$, the Σ-tree obtained by joining the Σ-trees $t_1, \ldots, t_n$ by a root node labeled by σ.

For every set Y we denote by

$$T_\Sigma Y$$

the algebra of all Σ-*trees over* Y, i.e., Σ-trees whose leaves are labeled by constant symbols in Σ_0 or elements of Y. This is the terminal coalgebra for $H_\Sigma(-) + Y$, and therefore it is a corecursive algebra.

▶ Remark 6. For a polynomial endofunctor H_Σ on Set we can view a coalgebra $e : X \to H_\Sigma X$ as a system of *recursive equations* over the set X of (recursion) variables: for every variable $x \in X$ we have a formal equation

$$x \approx \sigma(x_1, \ldots, x_n) = e(x).$$

The map $e^\dagger$ in Definition 4 is then a *solution* of the system of equations in the Σ-algebra A: the commutative square (1) states that $e^\dagger$ turns the above formal equations into actual identities in A: $e^\dagger(x) = \sigma^A(e^\dagger(x_1), \ldots, e^\dagger(x_n))$.

▶ **Definition 7** ([13])**.** An algebra $a : HA \to A$ is called *completely iterative* (or *cia*, for short) if the algebra $[a, A] : HA + A \to A$ is corecursive for the endofunctor $H(-) + A$. That means that for every *(flat) equation morphism* $e : X \to HX + A$ there exists a unique *solution*, i.e., a unique morphism $e^\dagger$ such that square below commutes:

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array} \tag{2}$$

▶ **Examples 8.**

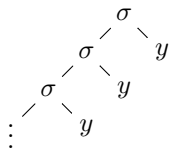
(1) If $H(-) + Y$ has a terminal coalgebra TY (cf. Example 5(1)), then (TY, τ_Y) is a cia. In fact, (TY, τ_Y) is a free cia on Y with the universal morphism η_Y [13].

(2) For a polynomial functor H_Σ on Set the above example states that the algebra $T_\Sigma Y$ of all Σ-trees over Y is the free cia on the set Y. Let us denote by

$$C_\Sigma Y$$

the subalgebra of $T_\Sigma Y$ given by all Σ-trees over Y which have only a finite number of leaves labeled in Y (and the remaining, possibly infinitely many, leaves are labeled in Σ_0). This algebra is corecursive but, whenever Σ contains an operation symbol of arity at least 2, not a cia. Moreover, $C_\Sigma Y$ is the free corecursive algebra on Y [3].

As a concrete example, consider the signature Σ consisting of a single binary operation σ. Then the equation morphism $e : \{x_1, x_2\} \to H_\Sigma\{x_1, x_2\} + \{y\}$ given by the recursive equations $x_1 \approx \sigma(x_1, x_2)$ and $x_2 \approx y$ has the unique solution $e^\dagger : \{x_1, x_2\} \to T_\Sigma\{y\}$ given as follows

$e^\dagger : x_1 \mapsto$  $x_2 \mapsto y.$

This demonstrates that $C_\Sigma\{y\}$ is not a cia because the above infinite Σ-tree is not contained in it.

▶ **Definition 9.** A *cia functor* is an endofunctor such that every corecursive algebra for it is a cia. (It the follows that cias and corecursive algebras coincide).

▶ **Notation 10.**

(1) If a free H-algebra on Y exists, we denote it by FY and its structure and universal morphism by

$$\varphi_Y : HFY \to FY \qquad \text{and} \qquad \eta^F_Y : Y \to FY,$$

respectively.

In the case of a polynomial set functor H_Σ, the free Σ-algebra $F_\Sigma Y$ is the subalgebra of $T_\Sigma Y$ on all finite Σ-trees over Y.

(2) If a free corecursive H-algebra on Y exists, we denote it by CY and its structure and universal morphism by

$$\psi_Y : HCY \to CY \qquad \text{and} \qquad \eta^C_Y : Y \to CY,$$

respectively.

3 Functors Preserving Countable Coproducts

▶ **Assumption 11.** In this and the subsequent section we assume that H is an endofunctor on a hyper-extensive category having a terminal coalgebra and preserving countable coproducts.

▶ **Fact 12** ([8]). *A free algebra on Y is*

$$FY = H^*Y = \coprod_{n<\omega} H^n Y \qquad \textit{with coproduct injections } j_n : H^n Y \to H^*Y.$$

Its algebra structure and universal morphism are given by

$$\varphi_Y \cdot Hj_n = j_{n+1} \quad (n > 0) \qquad \textit{and} \qquad \eta^F_Y = j_0 : Y \to H^*Y$$

using that $HFY = \coprod_{n<\omega} H^{n+1}Y$.

▶ **Notation 13.** We denote by

$$\sigma_Y : H^*Y = \coprod_{n<\omega} H^n Y \to Y + H\left(\coprod_{n<\omega} H^n Y\right) = Y + HH^*Y$$

the isomorphism inverse to $[\eta^F_Y, \varphi_Y] : Y + HH^*Y \to H^*Y$. It is defined by the following commutative diagrams:

$$\begin{array}{ccc} Y & & \\ {\scriptstyle j_0}\downarrow & \searrow^{\mathsf{inl}} & \\ H^*Y & \xrightarrow[\sigma_Y]{} & Y + HH^*Y \end{array} \qquad \begin{array}{ccc} H^n Y & \xrightarrow{Hj_{n-1}} & HH^*Y \\ {\scriptstyle j_n}\downarrow & & \downarrow{\scriptstyle \mathsf{inr}} \\ H^*Y & \xrightarrow[\sigma_Y]{} & Y + HH^*Y \end{array} \qquad \text{for } n > 0. \tag{3}$$

▶ **Theorem 14.** *The free cia on Y is*

$$CY = H^*Y + \nu H$$

with algebra structure $\varphi_Y + t^{-1} : H(H^*Y + \nu H) \cong HH^*Y + H(\nu H) \to H^*Y + \nu H$.

Proofsketch. In view of Example 8 it suffices to prove that the terminal coalgebra for $Y + H(-)$ is $H^*Y + \nu H$ with the following coalgebra structure

$$H^*Y + \nu H \xrightarrow{\sigma_Y + t} Y + HH^*Y + H(\nu H) \cong Y + H(H^*Y + \nu H).$$

That means that for a given coalgebra $e : X \to Y + HX$ there exists precisely one coalgebra morphism $h : X \to H^*Y + \nu H$. This morphism h is defined by an iterative construction using pullbacks and (hyper-)extensivity that we now explain.

Let $X_0 = X$ and $e_0 = e$ and denote the coproduct injections of $Y + HX$ by $i_0 : HX \to Y + HX$ and $\overline{i}_0 : Y \to Y + HX$. Next form the pullbacks of e along these injections:

$$\begin{array}{ccccc} X_1 & \xrightarrow{i_1} & X_0 & \xleftarrow{\overline{i}_1} & \overline{X}_1 \\ {\scriptstyle e_1}\downarrow & & \downarrow{\scriptstyle e_0} & & \downarrow{\scriptstyle \overline{e}_1} \\ HX & \xrightarrow[i_0]{} & Y + HX & \xleftarrow[\overline{i}_0]{} & Y \end{array} \tag{4}$$

By extensivity, $X = X_1 + \overline{X}_1$ with injections i_1 and $\overline{i}_1$. The component $\overline{h}_1 := h \cdot \overline{i}_1$ of h at $\overline{X}_1$ is defined by

$$h \cdot \overline{i}_1 = \left(\overline{X}_1 \xrightarrow{\overline{e}_1} Y \xrightarrow{j_0} H^*Y \xrightarrow{\mathsf{inl}} H^*Y + \nu H \right).$$

In order to analyze the complementary coproduct component $h \cdot i_1$, we form the pullbacks of e_1 along the coproduct injections of $HX_0 = HX_1 + H\overline{X}_1$:

$$\begin{array}{ccccc} X_2 & \xrightarrow{i_2} & X_1 & \xleftarrow{\overline{i}_2} & \overline{X}_2 \\ {\scriptstyle e_2}\downarrow & & \downarrow{\scriptstyle e_1} & & \downarrow{\scriptstyle \overline{e}_2} \\ HX_1 & \xrightarrow[Hi_1]{} & HX_0 & \xleftarrow[H\overline{i}_1]{} & H\overline{X}_1 \end{array}$$

Then $X_1 = X_2 + \overline{X}_2$ and the component $\overline{h}_2 = h \cdot i_1 \cdot \overline{i}_2$ of h at $\overline{X}_2$ is defined by

$$h \cdot i_1 \cdot \overline{i}_2 = \left(\overline{X}_2 \xrightarrow{\overline{e}_2} H\overline{X}_1 \xrightarrow{H\overline{e}_1} HY \xrightarrow{j_1} H^*Y \xrightarrow{\mathsf{inl}} H^*Y + \nu H \right).$$

We continue this process recursively: given a coproduct $X_n \xrightarrow{i_n} X_{n-1} \xleftarrow{\overline{i}_n} \overline{X}_n$ and a morphism $e_n : X_n \to HX_{n-1}$ we form its pullbacks along the coproduct injection of $HX_{n-1} = HX_n + H\overline{X}_n$:

$$\begin{array}{ccccc} X_{n+1} & \xrightarrow{i_{n+1}} & X_n & \xleftarrow{\overline{i}_{n+1}} & \overline{X}_{n+1} \\ {\scriptstyle e_{n+1}}\downarrow & & \downarrow{\scriptstyle e_n} & & \downarrow{\scriptstyle \overline{e}_{n+1}} \\ HX_n & \xrightarrow[Hi_n]{} & HX_{n-1} & \xleftarrow[H\overline{i}_n]{} & H\overline{X}_n \end{array} \tag{5}$$

Since compositions of coproduct injections are always coproduct injections, we obtain coproduct injections

$$\overline{i}^*_{n+1} = \left(\overline{X}_{n+1} \xrightarrow{\overline{i}_{n+1}} X_n \xrightarrow{i_n} X_{n+1} \xrightarrow{i_{n-1}} \cdots \xrightarrow{i_1} X \right) \qquad (n < \omega) \tag{6}$$

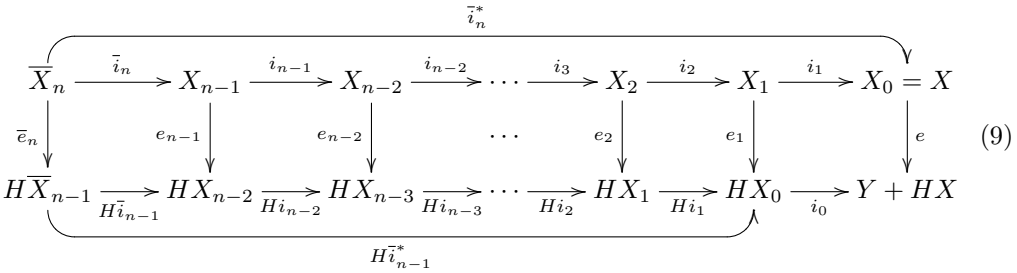
and morphisms

$$\widehat{e}_{n+1} = \left(\overline{X}_{n+1} \xrightarrow{\overline{e}_{n+1}} H\overline{X}_n \xrightarrow{H\overline{e}_n} H^2\overline{X}_{n-1} \xrightarrow{H^2\overline{e}_{n-1}} \cdots \xrightarrow{H^n\overline{e}_1} H^nY \right) \qquad (n < \omega). \tag{7}$$

The component $\overline{h}_{n+1} := (\overline{X}_{n+1} \xrightarrow{\overline{i}^*_{n+1}} X \xrightarrow{h} H^*Y + \nu H)$ of h at $\overline{X}_{n+1}$ is defined by the commutativity of the following square

$$\begin{array}{ccc} \overline{X}_{n+1} & \xrightarrow{\overline{i}^*_{n+1}} & X \\ {\scriptstyle \widehat{e}_{n+1}}\downarrow & & \downarrow{\scriptstyle h} \\ H^nY & \xrightarrow[j_n]{} H^*Y \xrightarrow[\mathsf{inl}]{} & H^*Y + \nu H \end{array} \tag{8}$$

Observe also that by composing pullback squares we obtain the following pullback:



(9)

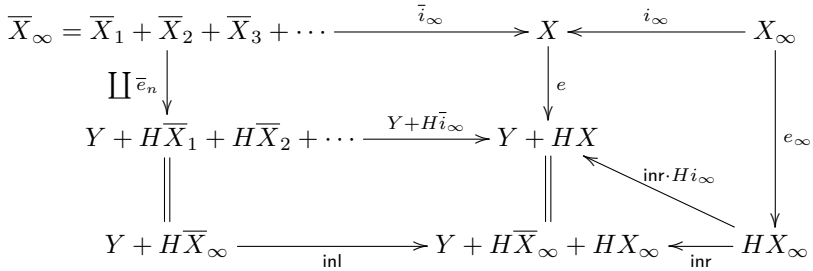
Now the coproduct injections in (6) are clearly pairwise disjoint. Therefore, by hyper-extensivity, we have a coproduct injection $[\overline{i}^*_{n+1}]_{n<\omega}$ which we denote by

$$\overline{X}_\infty \xrightarrow{\overline{i}_\infty} X \qquad \text{for} \qquad \overline{X}_\infty := \coprod_{n<\omega} \overline{X}_{n+1},$$

and $h \cdot \overline{i}_\infty$ is defined coponentwise by (8). By hyper-extensivity we can consider the complementary coproduct component $i_\infty : X_\infty \to X$, i.e., we have the coproduct

$$\overline{X}_\infty \xrightarrow{\overline{i}_\infty} X \xleftarrow{i_\infty} X_\infty.$$

Since the pullbacks (9) have pairwise disjoint coproduct injections as their upper arrows, they form together the pullback on the left below:



(10)

By extensivity, we obtain a morphism $e_\infty : X_\infty \to HX_\infty$ complementary to $\coprod \overline{e}_n$. This morphism is the structure of an H-coalgebra on X_∞. Thus, we define $h \cdot i_\infty$ to be the unique coalgebra morphism from X_∞ to νH.

One now verifies that the morphism $h : X \to H^*Y + \nu H$ so defined is a unique coalgebra morphism for $Y + H(-)$ as desired (see the full version [5] of our paper for details). ◀

▶ **Example 15.**

(a) It is well-known that the identity functor on Set has the free cias (equivalently, final coalgebras for $(-) + Y$) $TY = \mathbb{N} \times Y + 1$ where $\mathbb{N}$ is the set of natural numbers. It follows from Theorem 14 that the same formula holds in every hyper-extensive category with a terminal object 1. To see this, one first shows that

$$N := \coprod_{n<\omega} 1 \quad \text{with} \quad 1 \xrightarrow{\mathsf{in}_0} N \xleftarrow{[\mathsf{in}_{n+1}]_{n<\omega}} N$$

forms a natural number object, i.e., an initial algebra for $1 + (-)$. Using distributivity we see that for any object Y the free algebra Id^*Y is

$$\mathsf{Id}^*Y = \coprod_{n<\omega} Y \cong \left(\coprod_{n<\omega} 1\right) \times Y = N \times Y. \tag{11}$$

Finally, we clearly have $\nu\mathsf{Id} = 1$. By Theorem 14, we thus obtain

$$TY \cong N \times Y + 1.$$

(b) For the above formula giving the free cia for Id on every Y it is *not* sufficient that $\mathcal{C}$ be an extensive category. As a counterexample consider the category $\mathcal{C} = \mathsf{CHaus}$ of compact Hausdorff spaces. Its limits and finite coproducts are created by the forgetful functor into Set, thus CHaus is extensive. However, it is not hyper-extensive since countable coproducts are not universal. For $Y = 1$ (the one point space) the formula (11) gives an uncountable space since $\coprod_{n<\omega} 1$ is the Stone-Čech compactification of an infinite discrete space. However, in the notation of Example 5, $T1$ is a countable space; for the terminal ω^{op}-chain

$$1 \leftarrow 1 + 1 \leftarrow 1 + 1 + 1 \leftarrow \cdots$$

of the functor $\mathsf{Id} + 1$ on CHaus has the corresponding underlying chain in Set. The limit in Set is countable, giving the set $N + 1$. The limit in CHaus is then a compact space on this set, in fact, it is the one-point compactification of the discrete space on N. Since the functor $X \mapsto X + 1$ preserves this limit, it is its terminal coalgebra. That means that $T1$ is countable.

▶ **Example 16.** Extending Example 15(a), we know that the functor $HX = \Sigma \times X$ on Set has the free cias $TY = \Sigma^* \times Y + \Sigma^\omega$, where Σ^* and Σ^∞ are the usual sets of strings (words) and sequences (streams) on Σ.

It follows from Theorem 14 that the same formula holds in every hyper-extensive category $\mathcal{C}$ with finite products commuting with countable coproducts. Examples of such categories are presheaf categories, posets, graphs and unary algebras.

Given an object Σ of $\mathcal{C}$, the functor $HX = \Sigma \times X$ has the terminal coalgebra

$$\Sigma^\omega = \lim_{n<\omega} \Sigma^n$$

which is the limit of the ω^{op}-chain of projections as follows:

$$1 \xleftarrow{!} \Sigma \xleftarrow{\Sigma\times!} \Sigma \times \Sigma \xleftarrow{\Sigma\times\Sigma\times!} \Sigma \times \Sigma \times \Sigma \leftarrow \cdots$$

The free algebras H^*Y are obtained as follows: define

$$\Sigma^* = \coprod_{n<\omega} \Sigma^n.$$

Then $H^*Y = \Sigma^* \times Y$. Thus, according to Theorem 14, the free cia for H on Y is given by

$$TY = \Sigma^* \times Y + \Sigma^\omega.$$

Similarly, given another object W of $\mathcal{C}$, the functor $H'X = W + \Sigma \times X$ has the free cias $T'Y = \Sigma^* \times (W + Y) + \Sigma^\omega$.

▶ **Example 17.** In Theorem 14 it is not sufficient that H preserves finite coproducts. In fact, consider the ultrafilter functor $U : \mathsf{Set} \to \mathsf{Set}$ which assigns to every set X the set of all ultrafilters on X and to a map $f : X \to Y$ the map Uf sending an ultrafilter $\mathcal{A}$ on X to $\{B \subseteq Y \mid f^{-1}(B) \in \mathcal{A}\}$. It preserves finite coproducts and $\nu U = 1$. But for Y infinite, $Y + U(-)$ has no fixed points; for suppose that $TY \cong Y + UTY$, then TY must be infinite since Y is so and therefore $|TY| < |UTY|$ contradicting the isomorphism.

4 Corecursiveness vs. Complete Iterativity

Under Assumption 11 we prove in this section that H is a cia functor, i.e., every corecursive algebra is a cia. Let $a : HA \to A$ be a fixed algebra.

▶ **Notation 18.**

(1) Define morphisms

$$a^n : H^n A \to A$$

by the following induction:

$$a^0 = \mathsf{id}_A \qquad \text{and} \qquad a^{n+1} = (H^{n+1}A = HH^nA \xrightarrow{Ha^n} HA \xrightarrow{a} A).$$

(2) For every equation morphism $e : X \to HX + A$ we use the notation of the proof of Theorem 14, except that Y is replaced by A everywhere (and the order of summands is swapped). Thus we use the morphisms

$$i_n, \bar{i}_n, e_n, \bar{e}_n, e_\infty, i_\infty, \bar{i}_\infty, \widehat{e}_n, \text{ and, } \bar{i}_n^*$$

as in that proof.

▶ **Construction 19.** Let $a : HA \to A$ be an algebra. Given an equation morphism $e : X \to HX + A$ and a coalgebra-to-algebra morphism $s : X_\infty \to A$:

$$\begin{array}{ccc} X_\infty & \xrightarrow{s} & A \\ {\scriptstyle e_\infty}\downarrow & & \uparrow{\scriptstyle a} \\ HX_\infty & \xrightarrow[Hs]{} & HA \end{array} \tag{12}$$

we define a morphism $e_s^\dagger : X \to A$ on the components of the coproduct $X = \left(\coprod_{n\geq 1} \overline{X}_n\right) + X_\infty$ (with injections $\bar{i}_n^*$, for every $n \geq 1$, and i_∞) separately as follows:

$$\begin{array}{ccc} \overline{X}_n & \xrightarrow{\widehat{e}_n} & H^{n-1}A \\ {\scriptstyle \bar{i}_n^*}\downarrow & & \downarrow{\scriptstyle a^{n-1}} \\ X & \xrightarrow[e_s^\dagger]{} & A \end{array} \quad \text{for } n \geq 1, \text{ and } \quad \begin{array}{ccc} X_\infty & & \\ {\scriptstyle i_\infty}\downarrow & \searrow^{s} & \\ X & \xrightarrow[e_s^\dagger]{} & A \end{array} \tag{13}$$

▶ **Proposition 20.** *The morphism $e_s^\dagger$ is a solution of e. Moreover, every solution of e is of the form $e_s^\dagger$ for some coalgebra-to-algebra morphism s.*

▶ **Corollary 21.** *The functor H is a cia functor.*

Indeed, if (A, a) is a corecursive H-algebra and $e : X \to HX + A$ is a given equation morphism, we have a unique s as in (12). Now note that Proposition 20 establishes a bijective correspondence between solutions of e and coalgebra-to-algebra morphisms from e_∞ to a, and therefore there exists a unique solution of e.

▶ **Example 22.** For the ultrafilter functor U of Example 17 consider the subfunctor U_0 of all ω-complete ultrafilters, i.e., those closed under countable intersections. This functor preserves countable coproducts and $\nu U_0 = 1$. Assume that a proper class of measurable cardinals n

exists (i.e., for each n we have an ω-complete ultrafilter P on a set X not containing any subset of X of less than n elements). This is quite a strong assumption in set theory, but we make it here to derive a strong property of U_0: it is a non-accessible cia functor! Indeed, the latter follows from Corollary 21, and U_0 is not accessible: for every measurable cardinal n it does not preserve the n-filtered colimit of all subsets Y of X of cardinality less than n, since P lies in U_0X but not in U_0Y if $|Y| < n$. This is a surprising example in view of Theorem 37 which shows that such a complex example does not exist among finitary set functors.

Finally, note that both cias and corecursive algebras form full subcategories of the category of all algebras for H. Thus Corollary 21 establishes an isomorphism of categories between the categories of cias and corecursive algebras for H.

The following proposition needs no assumptions on H or the base category except that binary coproducts exist.

▶ **Proposition 23.** *If H is a cia functor, then so is $H(-) + Y$ for every object Y.*

▶ **Corollary 24.** *Let H be a functor having a terminal coalgebra and preserving countable coproducts. Then $H(-) + Y$ is a cia functor for every object Y.*

5 Finitary Set Functors

We have seen above that for every functor H on a hyper-extensive category preserving countable coproducts, the functors $H(-) + Y$ are cia functors (i.e., every corecursive algebra is a cia). In particular, if $\mathcal{C}$ is cartesian closed, then the functor $X \mapsto W \times X + Y$ is a cia functor. For $\mathcal{C} = \mathsf{Set}$ and H finitary we now prove the converse: if H is a cia functor then it has the form $X \mapsto W \times X + Y$ for some sets W and Y.

▶ **Assumption 25.** Throughout this section H denotes a standard, finitary set functor.

Recall from [6] that H is *finitary* iff for every set X we have $HX = \bigcup HY$ where the union ranges over finite subsets $Y \subseteq X$. An example of a finitary functor on Set is the polynomial functor H_Σ, see Example 5(3).

Standard means that H preserves

(1) inclusions, i.e., $X \subseteq Y$ implies $HX \subseteq HY$ and the H-image of the inclusion map $X \hookrightarrow Y$ is the inclusion map $HX \hookrightarrow HY$, and

(2) finite intersections.

Assuming that H is standard is without loss of generality because for every set functor H there exist a standard set functor H' naturally isomorphic to H on the full subcategory of all nonempty sets [7, Theorem 3.4.5]. (And the change of value at $\emptyset$ is irrelevant for us since corecursive algebras and cias, respectively, for H are in bijective correspondence with those for H').

▶ **Definition 26.**

(1) By a *presentation* of H is meant a finitary signature Σ and natural epitransformation $\varepsilon : H_\Sigma \to H$, i.e., every component ε_X is a surjective map.

(2) An *ε-equation* is an expression $\sigma(x_1, \dots x_n) = \tau(z_1, \dots, z_m)$ where σ is an n-ary operation symbol and τ an m-ary one such that ε_X merges the two elements of $H_\Sigma X$ where $X = \{x_1, \dots, x_n, z_1, \dots, z_n\}$.

▶ Remark 27. All ε-equations form an equivalence relation. More precisely, for any set X all ε-equations with variables replaced by elements of X form precisely the kernel equivalence of ε_X. Moreover, the elements of HX may be regarded as equivalence classes of the elements $\sigma(x_1, \dots, x_n)$ of $H_\Sigma X$ modulo this equivalence.

▶ **Example 28.** The finite power-set functor $\mathcal{P}_\mathsf{f}$ has a presentation with Σ having a single n-ary operation for every n, and ε sending $\sigma(x_1,\dots,x_n)$ to $\{x_1,\dots,x_n\}$.

The following lemma was proved in [7]. We present a (short) proof since we refer to it later.

▶ **Lemma 29.** *Every finitary set functor has a presentation $\varepsilon : H_\Sigma \to H$, and the category* $\mathsf{Alg}\, H$ *is isomorphic to the variety of all Σ-algebras satisfying all ε-equations.*

Proof. Define a signature $\Sigma = (\Sigma_n)_{n<\omega}$ by $\Sigma_n = Hn$ where we regard n as the finite ordinal $\{0,\dots,n-1\}$ for all n. By the Yoneda lemma we have a natural transformation $\varepsilon_X : H_\Sigma X \to HX$ assigning to every $\sigma(x_1,\dots,x_n)$ represented as a function $x : n \to X$ the element $Hx(\sigma)$. Since H is finitary, ε_X is surjective.

Every H-algebra $a : HA \to A$ defines the corresponding Σ-algebra $a \cdot \varepsilon_A : H_\Sigma A \to A$ which clearly satisfies all ε-equations. This defines a full embedding of $\mathsf{Alg}\, H$ into $\mathsf{Alg}\, H_\Sigma$ (which is identity on morphisms). We now easily prove that every Σ-algebra satisfying all ε-equations has the above form $(A, a \cdot \varepsilon_A)$. Indeed, given $a^\Sigma : H_\Sigma A \to A$ satisfying all ε-equations, define $a : HA \to A$ by $a([\sigma(a_1,\dots,a_n]) = a^\Sigma(\sigma(a_1,\dots,a_n))$. Since we know from Remark 27 that a^Σ merges all pairs in the kernel of ε_A, this is well-defined and we clearly have $a^\Sigma = a \cdot \varepsilon_A$. Thus, our full embedding defines the desired isomorphism between H-algebras and Σ-algebras satisfing all ε-equations. ◀

▶ Remark 30.

(1) Denote by C_1 the constant functor with value $1 = \{c\}$, and by $C_{0.1}$ its subfunctor with $C_{0,1}\emptyset = \emptyset$ and $C_{0,1}X = 1$ else. For every natural transformation $\alpha : C_{0,1} \to H$ there exists a unique extension to $\alpha' : C_1 \to H$.
Indeed, since H is standard, it preserves the (empty) intersection of the coproduct injections $\mathsf{inl}, \mathsf{inr} : 1 \to 1+1$. Since $H\mathsf{inl}(\alpha_1(c)) = \alpha_{1+1}(c) = H\mathsf{inr}(\alpha_1(c))$, there exists a unique element t of $H\emptyset$ such that the inclusion map $v : \emptyset \to 1$ fulfils $\alpha_1(c) = Hv(t)$. We put $\alpha'_\emptyset(c) = t$.

(2) All constants in our presentation of H are *explicit*. That means that whenever some n-ary symbol σ has the property that some ε-equation has the form $\sigma(x_1,\dots,x_n) = \sigma(z_1,\dots,z_n)$, where the variables x_i are pairwise distinct and none of them equals some z_j, then there exists a constant symbol τ in Σ for which we have the following ε-equation: $\sigma(x_1,\dots,x_n) = \tau$. Indeed, for every set $X \neq \emptyset$ we have an element

$$\alpha_X = \varepsilon_X(\sigma(a_1,\dots,a_n)) \in HX$$

independent of the choice of $a_1,\dots,a_n$ in X. This defines a natural transformation $\alpha : C_{0,1} \to H$. Let $\alpha' : C_1 \to H$ be its extension according to item (1). The element $\alpha'_\emptyset(c)$ of $H\emptyset$ has, since ε is an epitransformation, the form $\varepsilon_\emptyset(\tau)$ for some nullary symbol τ. Then the desired ε-equation holds because for $X = \{x_1,\dots,x_n\}$ and the unique empty map $u : \emptyset \to X$ we have

$$\varepsilon_X(\sigma(x_1,\dots,x_n)) = \alpha_X(c) = \alpha'_X(c) = Hu \cdot \alpha'_\emptyset(c) = Hu \cdot \varepsilon_\emptyset(\tau) = \varepsilon_X \cdot Hu(\tau) = \varepsilon_X(\tau).$$

▶ **Definition 31.** A presentation $\varepsilon : H_\Sigma \to H$ is *reduced* provided that for every ε-equation

$$\sigma(x_1,\dots,x_n) = \tau(z_1,\dots,z_m)$$

the following hold:

(1) if $x_1,\dots,x_n$ are pairwise distinct, then they all lie in $\{z_1,\dots,z_n\}$, and

(2) if, moreover, $z_1,\dots,z_n$ are also pairwise distinct, then $\sigma = \tau$.

▶ **Proposition 32.** *Every finitary set functor has a reduced presentation.*

▶ **Notation 33.** From now on we assume that a reduced presentation of H is given.

Recall the notation TY, FY and CY from Examples 8 and Notation 10. All these objects exist since H is finitary (and therefore so are all $H(-)+Y$). The corresponding notation for H_Σ is $T_\Sigma Y$, $F_\Sigma Y$ and $C_\Sigma Y$. The monad units of T and C are denoted by η and η^C, respectively.

As mentioned above, $T_\Sigma Y$ can be described as the algebra of all Σ-trees over Y. And $C_\Sigma Y$ and $F_\Sigma Y$ are its subalgebras on all trees with finitely many leaves labeled in Y, or all finite trees, respectively.

Since TY is a corecursive algebra, there exists a unique homomorphism of H-algebras

$$m_Y : CY \to TY$$

with $m_Y \cdot \eta^C_Y = \eta_Y$. The corresponding H_Σ-algebra morphism is denoted by

$$m^\Sigma_Y : C_\Sigma Y \to T_\Sigma Y.$$

▶ Remark 34. In [4] we described FY and TY as the following quotient of the Σ-algebras $F_\Sigma Y$ and $T_\Sigma Y$, respectively. Recall from Lemma 29 that every H-algebra $a : HA \to A$ may be regarded as the H_Σ-algebra with structure $a \cdot \varepsilon_A : H_\Sigma A \to A$.

(1) $FY = F_\Sigma Y/{\sim_Y}$, where $\sim_Y$ is the congruence of finite application of ε-equations. That is, the smallest congruence with $\sigma(x_1, \ldots, x_n) \sim_Y \tau(z_1, \ldots, z_m)$ for every ε-equation

$$\sigma(x_1, \ldots, x_n) = \tau(z_1, \ldots, z_m)$$

over Y. The universal map $\eta^F_Y : Y \to FY$ is the composition of the one of $F_\Sigma Y$ with the canonical quotient map $F_\Sigma Y \twoheadrightarrow F_\Sigma Y/{\sim_Y}$.

(2) $TY = T_\Sigma Y/{\sim^*_Y}$, where $\sim^*_Y$ is the congruence of (possibly infinitely many) applications of ε-equations. The universal map is $\widehat{\eta}_Y = \widehat{\varepsilon}_Y \cdot \eta^\Sigma_Y$, where $\eta^\Sigma_Y : Y \to T_\Sigma Y$ is the universal map of the free cia for H_Σ on Y and $\widehat{\varepsilon}_Y : T_\Sigma Y \twoheadrightarrow T_\Sigma Y/{\sim^*_Y}$ is the canonical quotient map.

The definition of a possibly *infinite application of ε-equations* is based on the concept of *cutting* a Σ-tree at level k: the resulting finite Σ-tree $\partial_k t$ is obtained from t by deleting all nodes of depth larger than k and relabeling all nodes at level k by a symbol $\bot \notin Y$. Then we define, for Σ-trees t and s in $T_\Sigma Y$,

$$t \sim^*_Y s \qquad \text{iff} \qquad \partial_k t \sim_{Y \cup \{\bot\}} \partial_k s \quad \text{for every } k < \omega.$$

Not surprisingly, CY can be described analogously:

▶ **Proposition 35.** *The free corecursive H-algebra CY is the quotient of the Σ-algebra $C_\Sigma Y$ modulo the application of ε-equations: $CY = C_\Sigma Y/{\sim^*_Y}$.*

Proof. This is based on the following description of CY presented in [3]: denote by $\oplus$ the binary coproduct of H-algebras in $\mathsf{Alg}\, H$. By Lemma 29, this is, equivalently, the coproduct in the variety of all Σ-algebras satisfying all ε-equations. Then we have

$$CY = \nu H \oplus FY.$$

Analogously, if $\boxplus$ denotes the binary coproduct of Σ-algebras, we of course have

$$C_\Sigma Y = \nu H_\Sigma \boxplus F_\Sigma Y.$$

For arbitrary H-algebras A and B we know that $A \oplus B$ is the quotient of $A \boxplus B$ modulo the application of ε-equations. Moreover, we have $T = T_\Sigma/{\sim^*}$ and $FY = F_\Sigma Y/{\sim}$. It follows immediately that $T \oplus FY = (T_\Sigma \boxplus F_\Sigma Y)/{\sim^*}$, as claimed. ◀

▶ **Lemma 36.** *Suppose that CY is a cia for H. For every equation morphism $e : X \to H_\Sigma X + Y$ with the unique solution $e^\ddagger : X \to T_\Sigma Y$ we can form an equation morphism*

$$\overline{e} = (X \xrightarrow{e} H_\Sigma X + Y \xrightarrow{\varepsilon_X + \eta^C_Y} HX + CY).$$

Then we have $(X \xrightarrow{\overline{e}^\dagger} CY \xrightarrow{m_Y} TY) = (X \xrightarrow{e^\ddagger} T_\Sigma Y \xrightarrow{\widehat{\varepsilon}_Y} TY)$.

▶ **Theorem 37.** *For a finitary set functor H the following conditions are equivalent:*

(1) *H is a cia functor,*

(2) *$H = H_0(-) + Y$ where H_0 preserves countable coproducts and Y is a set, and*

(3) *$H = W \times (-) + Y$ for some sets W and Y.*

Proof.

(2) ⇒ (3). Since H is finitary, so is H_0, by the description of finitarity following Assumptions 25. Therefore, H_0 preserves all coproducts. Trnková proved [16, Theorem IX.8], that every coproduct-preserving set functor preserves colimits, thus it is a left adjoint. It is well known that the only right adjoint set functors R are the representable ones: for given $L \dashv R$, put $W = L1$, then the elements $1 \to RY$ bijectively correspond to the maps $W \to Y$, thus, R is naturally isomorphic to $\mathsf{Set}(W, -)$. Consequently, H_0 is left adjoint to $\mathsf{Set}(W, -)$, hence it is naturaly isomorphic to $W \times (-)$.

(3) ⇒ (1). This follows from Corollary 24.

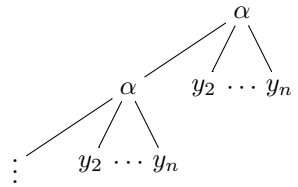
(1) ⇒ (2). Let $\varepsilon : H_\Sigma \to H$ be a reduced presentation.

(a) We prove below that all arities in Σ are 1 or 0. Let W be the set of all unary symbols and Y that of all constants. Then $H_\Sigma X = W \times X + Y$. Furthermore, we show that ε is a natural isomorphism. Indeed, each ε_X is, besides being surjective, also injective: it cannot merge distinct elements (w, x) and (w', x') of $W \times X$ because this would yield an ε-equation $w(x) = w'(x')$. Since the presentation is reduced, this implies $w = w'$ and $x = x'$. Analogously for all other pairs of elements of $H_\Sigma X$.

(b) Assume that some symbol α of Σ has arity at least 2. Then we derive a contradiction to H being a cia functor. Given a Σ-tree t we call a node r *pure* if the trees $t_1, \ldots, t_n$ rooted at the children of r are pairwise distinct. Observe that an ε-equation applicable to a pure node r must have the form $\sigma(x_1, \ldots, x_n) = \tau(y_1, \ldots, y_m)$ for some $\tau \in \Sigma_m$, where $x_1, \ldots, x_n$ are pairwise distinct.

Consider the following equation morphism $e : X \to H_\Sigma X + Y$ with $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_2, \ldots, y_n\}$: $e(x_1) = \alpha(x_1, y_2, y_n)$ and $e(x_i) = y_i$, for $i = 2, \ldots, n$. Then the unique solution $e^\ddagger : X \to T_\Sigma Y$ assigns to x_1 the Σ-tree below:

$e^\ddagger(x_1) =$



Next consider the equation morphism $\overline{e} = (X \xrightarrow{e} H_\Sigma X + Y \xrightarrow{\varepsilon_X + \eta^C_Y} HX + CY)$. Since CY is a cia, this has a unique solution $e^\dagger : X \to CY$. It assigns to x_1 an element of CY which by Proposition 35 has the form $\overline{e}^\dagger(x_1) = \overline{\varepsilon}_Y(s)$ for some $s \in C_\Sigma Y$, where $\overline{\varepsilon}_Y : C_\Sigma Y \twoheadrightarrow C_\Sigma Y/{\sim^*} \cong CY$ denotes the canonical quotient map. From Lemma 36 we know that

$$\widehat{\varepsilon}_Y(t) = \widehat{\varepsilon}_Y \cdot e^\ddagger(x_1) = m_Y \cdot \overline{e}^\dagger(x_1) = m_Y \cdot \overline{\varepsilon}_Y(s).$$

Therefore, we obtain $t \sim_Y^* s$.

We derive the desired contradiction by proving that every tree obtained from t by a finite application of ε-equations has a leaf labeled by y_2 at every positive level. From this we conclude immediately that the same holds for all trees obtained by an infinite application of ε-equations from t. However, $t \sim_Y^* s$ where s has only finitely many leaves labeled by y_2.

(b1) Assume that a single ε-equation is applied to t and let t' be the resulting tree. Let r be the node of t at which the application takes place. Then r is not a leaf labeled in Y; for recall that all ε-equations have operation symbols on both sides, thus, they are not applicable to leaves labeled in Y. Therefore, r is a pure node labeled by α. The ε-equation in question thus has the form $\alpha(u_1, \ldots, u_n) = \tau(z_1, \ldots, z_m)$ for some $\tau \in \Sigma_m$ and with the u_i pairwise distinct.

If r has depth k, then the tree t' has label y_2 at all levels $1, \ldots, k$, since those leaves of t are unchanged. Furthermore, we have $u_2 = z_p$ for some $p = 1, \ldots, m$ since ε is a reduced presentation. Therefore, y_2 occurs at level $k+1$ since the p-th child of r in t' is a leaf labeled by y_2. For the levels greater than $k+1$ we use that $u_1 = z_q$ holds for some $q = 1, \ldots, m$, again because ε is a reduced presentation. Since the first subtree of r in t is t itself, it follows that the q-th child of r in t' is t itself. Thus, a label y_2 of depth n in t yields a label y_2 of depth $k+1+n$ of t'.

(b2) Assume that two ε-equations are applied to t. The resulting tree t'' can be obtained from t' in (b1) by a single application of an ε-equation. Let r' be the node of t' at which the application takes place. We can assume $r \neq r'$ (for if $r = r'$ we can obtain t'' from t by a single application on an ε-equation; this follows from Remark 27). If r' does not lie in the subtree of t' with root r, then r' is a pure node labeled by α and we argue as in (b1).

Suppose therefore that r' lies in the subtree rooted at r. If this is the q-th subtree from (b1) above (the one with $u_1 = z_q$), then we also argue as in (b1) using that the q-th subtree is t itself. Otherwise, if r' lies in any other subtree of r, then the labels y_2 of the q-th subtree are unchanged.

The remaining cases of three and more applications of ε-equations are completely analogous. This yields the desired contradiction: if $t \sim^* \bar{t}$, then $\bar{t}$ has label y_2 at every level $1, 2, 3, \ldots$, thus $t \sim_Y^* s$ cannot be true. ◀

6 Conclusions and Open Problems

For endofunctors H preserving countable coproducts and having a terminal coalgebra we have described the free corecursive algebra on an object Y as $\nu H + \coprod_{n<\omega} H^n Y$. In addition, we have shown that H is a cia functor, i.e., every corecursive algebra for H is a cia. For this we assumed that the base category has well-behaved countable coproducts, i.e., the category is hyper-extensive. It is an open problem whether our results hold in more general categories, e.g., in all extensive locally presentable ones.

For accessible functors H on locally presentable categories, the free corecursive algebra on Y was described in previous work [3] as the coproduct of FY (the free algebra on Y) and νH (considered as an algebra) in the category $\mathsf{Alg}\, H$. If H preserves countable coproducts, this is quite similar to the above desciption of the free cia, since coproducts of algebras are then formed on the level of the underlying category and therefore $FY = \coprod_{n<\omega} H^n Y$. But the proof techniques are completely different, and a common generalization of the two results is open.

We have also characterized all cia functors among finitary set functors: they are precisely the functors $X \mapsto W \times X + Y$ for some sets W and Y. In Example 22 we have seen that the same result does not hold for all, not necessarily finitary, set functors. But that example required an assumption about set theory. It is an open problem whether that assumption was really necessary.

Our results can be stated in terms of corecursive monads [3] and completely iterative ones [1] as follows: a functor H having a terminal coalgebra νH and preserving countable coproducts has a free corecursive monad of the form $\coprod_{n<\omega} H^n(-) + \nu H$, and this is also the free completely iterative monad on H.

References

1 Peter Aczel, Jiří Adámek, Stefan Milius, and Jiří Velebil. Infinite trees and completely iterative theories: A coalgebraic view. *Theoret. Comput. Sci.*, 300:1–45, 2003. Fundamental study.

2 Jiří Adámek, Reinhard Börger, Stefan Milius, and Jiří Velebil. Iterative algebras: How iterative are they? *Theory Appl. Categ.*, 19:61–92, 2008.

3 Jiří Adámek, Mahdieh Haddadi, and Stefan Milius. Corecursive algebras, corecursive monads and Bloom monads. *Log. Methods Comput. Sci.*, 10(3:19):51 pp., 2014.

4 Jiří Adámek and Stefan Milius. Terminal coalgebras and free iterative theories. *Inform. and Comput.*, 204:1139–1172, 2006.

5 Jiří Adámek and Stefan Milius. On corecursive algebras for functors preserving coproducts. full version, 2017. URL: `https://arxiv.org/abs/1703.07574`.

6 Jiří Adámek and Hans-Eberhard Porst. On tree coalgebras and coalgebra presentations. *Theoret. Comput. Sci.*, 311:257–283, 2004.

7 Jiří Adámek and Věra Trnková. *Automata and Algebras in Categories*, volume 37 of *Mathematics and its Applications*. Kluwer Academic Publishers, 1990.

8 Michael A. Arbib and Ernest G. Manes. Foundations of system theory: Decomposable systems. *Automatica*, 10:285–302, 1974.

9 Venanzio Capretta, Tarmo Uustalu, and Varmo Vene. Corecusive algebras: A study of general structured corecursion. In M. V. M. Oliveira and J. Woodcock, editors, *Proc. Brazilian Symposium on Formal Methods (SBMF'09)*, volume 5902 of *Lecture Notes Comput. Sci.*, pages 84–100. Springer, 2009.

10 Aurelio Carboni, Steve Lack, and Robert F. C. Walters. Introduction to extensive and distributive categories. *J. Pure Appl. Algebra*, 84:145–158, 1993.

11 Calvin C. Elgot. Monadic computation and iterative algebraic theories. In H. E. Rose and J. C. Sheperdson, editors, *Logic Colloquium '73*, volume 80, pages 175–230, Amsterdam, 1975. North-Holland Publishers.

12 Joachim Lambek. A fixpoint theorem for complete categories. *Math. Z.*, 103:151–161, 1968.

13 Stefan Milius. Completely iterative algebras and completely iterative monads. *Inform. and Comput.*, 196:1–41, 2005.

14 Evelyn Nelson. Iterative algebras. *Theoret. Comput. Sci.*, 25:67–94, 1983.

15 Jerzy Tiuryn. Unique fixed points vs. least fixed points. *Theoret. Comput. Sci.*, 12:229–254, 1980.

16 Věra Trnková. Descriptive classification of set functors ii. *Comment. Math. Univ. Carolin.*, 12:345–357, 1971.

Bisimulation for Weakly Expressive Coalgebraic Modal Logics

Zeinab Bakhtiari[*1] **and Helle Hvid Hansen**[2]

1 **LORIA, CNRS-Université de Lorraine, France**
bakhtiarizeinab@gmail.com
2 **Delft University of Technology, Delft, The Netherlands**
h.h.hansen@tudelft.nl

Abstract

Research on the expressiveness of coalgebraic modal logics with respect to semantic equivalence notions has so far focused mainly on finding logics that are able to distinguish states that are not behaviourally equivalent (such logics are said to be expressive). In other words, the notion of behavioural equivalence is taken as the starting point, and the expressiveness of the logic is evaluated against it. However, for some applications, modal logics that are not expressive are of independent interest. Such an example is given by contingency logic. We can now turn the question of expressiveness around and ask, given a modal logic, what is a suitable notion of semantic equivalence? In this paper, we propose a notion of Λ-bisimulation which is parametric in a collection Λ of predicate liftings. We study the basic properties of Λ-bisimilarity, and prove as our main result a Hennessy-Milner style theorem, which shows that (for finitary functors) Λ-bisimilarity exactly matches the expressiveness of the coalgebraic modal logic arising from Λ.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages, F.4.1 Mathematical Logic, I.2.4 Knowledge Representation Formalisms and Methods

Keywords and phrases Coalgebraic modal logic, bisimulation, expressiveness, Hennessy-Milner theorem

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.4

1 Introduction

Coalgebraic modal logic, as in [21, 13], is a framework in which modal logics for specifying coalgebras can be developed parametric in the signature of the modal language and the coalgebra type functor T. Given a base logic (usually classical propositional logic), modalities are interpreted via so-called predicate liftings for the functor T. These are natural transformations that turn a predicate over the state space X into a predicate over TX. Given that T-coalgebras come with general notions of T-bisimilarity [23] and behavioral equivalence [14], coalgebraic modal logics are designed to respect those. In particular, if two states are behaviourally equivalent then they satisfy the same formulas. If the converse holds, then the logic is said to be expressive. and we have a generalisation of the classic Hennessy-Milner theorem [9] which states that over the class of image-finite Kripke models, two states are Kripke bisimilar if and only if they satisfy the same formulas in Hennessy-Milner logic.

General conditions for when an expressive coalgebraic modal logic for T-coalgebras exists have been identified in [22, 3, 24]. A condition that ensures that a coalgebraic logic is

* Zeinab Bakthiari was funded by ERC grant EPS 313360.

expressive is when the set of predicate liftings chosen to interpret the modalities is *separating* [22]. Informally, a collection of predicate liftings is separating if they are able to distinguish non-identical elements from TX. This line of research in coalgebraic modal logic has thus taken as starting point the semantic equivalence notion of behavioral equivalence (or T-bisimilarity), and provided results for how to obtain an expressive logic. However, for some applications, modal logics that are not expressive are of independent interest. Such an example is given by *contingency logic* (see e.g. [6, 19]). We can now turn the question of expressiveness around and ask, given a modal language, what is a suitable notion of semantic equivalence?

In this paper, we propose a notion of Λ-bisimulation which is parametric in a collection Λ of predicate liftings, and therefore tailored to the expressiveness of a given coalgebraic modal logic. The definition relies on the notion of Z-coherent pairs, where Z is a relation between the state spaces of the relevant coalgebras. Coherent pairs were introduced in [8] when studying coalgebraic semantic equivalence notions in neighbourhood frames. In particular, we see that if T is the neighbourhood functor and Λ consists of the usual neighbourhood modality, then Λ-bisimulation amounts to the notion of precocongruence for neighbourhood frames from [8]. We observe that coherent pairs have an abstract characterisation in terms of pullbacks and pushouts which makes it possible to prove most of our results using general category theoretical arguments. This suggests to us that Λ-bisimulations are a natural concept, which may be useful when considering coalgebraic modal logics over other categories than Sets. Moreover, we show that Λ-bisimulations, like T-bisimulations, form a complete lattice, and we show how they relate to T-bisimulations, behavioural equivalences and precocongruences. We also discuss their relationship to similar notions proposed by Gorin & Schröder [7] and Enqvist [4]. Our main result is a finitary Hennessy-Milner theorem (which does not assume Λ is separating): If T is finitary, then two states are Λ-bisimilar if and only if they satisfy the same modal Λ-formulas.

Overview. In Section 2 we fix notation and introduce the notion of coherent sets. In Section 3 we define our notion of Λ-bisimulation, study its properties, and relate it to other existing equivalence notions. Our Hennessy-Milner theorem is proved in Section 4. The paper concludes with a discussion of future and related work in Section 5.

2 Preliminaries

We will work in the category Sets of sets and functions. The contravariant powerset functor $Q\colon \mathsf{Sets} \to \mathsf{Sets}^{\mathrm{op}}$ sends a set X to the powerset of X and a function $f\colon X \to Y$ to the inverse image map $Qf = f^{-1}\colon QY \to QX$. We assume familiarity with basic coalgebraic concepts and only provide the basic definitions. For an introduction, we refer to [23]. Given a functor $T\colon \mathsf{Sets} \to \mathsf{Sets}$, a T-coalgebra is a pair $(X, \gamma\colon X \to TX)$. A T-coalgebra morphism from (X, γ) to (Y, δ) is a function $f\colon X \to Y$ such that $Tf \circ \gamma = \delta \circ f$.

2.1 Coalgebraic modal logic

Coalgebraic modal logic [21] is a uniform framework in which modal logics for coalgebras can be developed parametric in the type functor T and a choice of predicate lifting.

Syntax. Given a *similarity type* Λ, which is a set of modal operators with finite arities, we define the syntax of coalgebraic modal logic as follows.

▶ **Definition 1.** The set $\mathcal{L}_\Lambda$ of Λ-formulas is generated by the following grammar:

$$\mathcal{L}_\Lambda \ni \varphi \quad ::= \quad \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \heartsuit(\underbrace{\varphi, \ldots, \varphi}_{n \text{ times}}) \quad (\heartsuit \in \Lambda, n\text{-ary})$$

We use the standard definitions of the Boolean operators $\bot, \vee$ and $\rightarrow$.

A T-coalgebraic semantics of $\mathcal{L}_\Lambda$-formulas is given by providing a Λ-*structure* $(T, (\llbracket\heartsuit\rrbracket)_{\heartsuit\in\Lambda})$ where T is a functor on Sets, and for each n-ary $\heartsuit \in \Lambda$, $\llbracket\heartsuit\rrbracket$ is an n-ary predicate lifting, i.e., $\llbracket\heartsuit\rrbracket : Q^n \Rightarrow QT$ is a natural transformation. Different choices of predicate liftings yield different Λ-structures and consequently different logics.

Semantics. Given a Λ-structure $(T, (\llbracket\heartsuit\rrbracket)_{\heartsuit\in\Lambda})$, the truth of $\mathcal{L}_\Lambda$-formulas in a T-coalgebra $\mathbb{X} = (X, \gamma\colon X \to TX)$ is defined as follows:

$$\begin{array}{lll} \mathbb{X}, x \models \top & & \text{always} \\ \mathbb{X}, x \models \neg\varphi & \text{iff} & X, x \not\models \varphi \\ \mathbb{X}, x \models \varphi \wedge \psi & \text{iff} & \mathbb{X}, x \models \varphi \text{ and } \mathbb{X}, x \models \psi \\ \mathbb{X}, x \models \heartsuit(\varphi_1, \ldots, \varphi_n) & \text{iff} & \gamma(x) \in \llbracket\heartsuit\rrbracket_X(\llbracket\varphi_1\rrbracket_{\mathbb{X}}, \ldots, \llbracket\varphi_n\rrbracket_{\mathbb{X}}). \end{array}$$

where $\llbracket\varphi\rrbracket_{\mathbb{X}} = \{x \in X \mid \mathbb{X}, x \models \varphi\}$ for all $\varphi \in \mathcal{L}_\Lambda$. Two states x in $\mathbb{X}$ and y in $\mathbb{Y}$ are modally equivalent (notation: $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$), if they satisfy the same $\mathcal{L}_\Lambda$-formulas, i.e., $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$ if for all $\varphi \in \mathcal{L}_\Lambda$, $\mathbb{X}, x \models \varphi$ iff $\mathbb{Y}, y \models \varphi$.

Pattinson in [22] introduced the notion of a *separating* set of predicate liftings when studying expressive logics.

▶ **Definition 2.** A set $(\llbracket\heartsuit\rrbracket)_{\heartsuit\in\Lambda}$ of predicate liftings for a functor T is *separating (for T)* if every $t \in TX$ is uniquely determined by the set $\{((A_1, ..., A_n), \heartsuit) \in (\mathcal{P}X)^n \times \Lambda \mid t \in \llbracket\heartsuit\rrbracket_X(A_1, ..., A_n)\}$. That is, if $t_1, t_2 \in TX$ and $t_1 \neq t_2$ then there is an n-ary $\heartsuit \in \Lambda$ and $A_1, \ldots, A_n \in \mathcal{P}X$ such that $t_1 \in \llbracket\heartsuit\rrbracket(A_1, \ldots, A_n)$ and $t_2 \notin \llbracket\heartsuit\rrbracket(A_1, \ldots, A_n)$, or vice versa.

We provide some examples of modal languages and their coalgebraic semantics.

▶ **Example 3.** Coalgebras for the covariant powerset functor $\mathcal{P}$ are Kripke frames. The similarity type $\Lambda = \{\Box\}$ for the basic modal language (without proposition letters) is given the usual Kripke semantics by interpreting $\Box$ via the predicate lifting $\llbracket\Box\rrbracket_X(A) = \{B \in \mathcal{P}X \mid B \subseteq A\}$, which is separating for $\mathcal{P}$, cf. [22].

Proposition letters can be included in the language by interpreting them as nullary predicate liftings. More precisely, given a set AtProp of proposition letters, the basic modal language over AtProp is obtained from the similarity type $\Lambda = \{\Box\} \cup \mathsf{AtProp}$, This language is given its usual semantics in Kripke models which are coalgebras for the functor $TX = \mathcal{P}(X) \times \mathcal{P}(\mathsf{AtProp})$ by taking the Λ-structure $(T, (\llbracket\heartsuit\rrbracket)_{\heartsuit\in\Lambda})$ where $\llbracket\Box\rrbracket_X(A) = \{(B, P) \in \mathcal{P}(X) \times \mathcal{P}(\mathsf{AtProp}) \mid B \subseteq A\}$ and $\llbracket p\rrbracket_X(A) = \{(B, P) \in \mathcal{P}(X) \times \mathcal{P}(\mathsf{AtProp}) \mid p \in P\}$.

▶ **Example 4.** The language of *contingency logic* [6] corresponds to the modal similarity type $\Lambda = \{\Delta\}$ and it is interpreted over Kripke frames (i.e. $\mathcal{P}$-coalgebras) via the predicate lifting $\llbracket\Delta\rrbracket_X(A) = \{B \in \mathcal{P}X \mid B \subseteq A \text{ or } B \subseteq A^c\}$. It is straightforward to check that $\llbracket\Delta\rrbracket$ is not separating for $\mathcal{P}$.

▶ **Example 5.** Neighbourhood frames are coalgebras for the functor $\mathcal{N} = Q^{\mathrm{op}}Q$. We obtain the neighbourhood semantics of the basic modal language, where $\Lambda = \{\Box\}$, by taking $\llbracket\Box\rrbracket_X(A) = \{B \in \mathcal{N}X \mid A \in B\}$, which is separating for $\mathcal{N}$.

▶ **Example 6.** Neighbourhood semantics of contingency logic [5] is obtained by taking $T = \mathcal{N}$, $\Lambda = \{\Delta\}$, and $[\![\Delta]\!]_X(X) = \{B \in \mathcal{N}X \mid A \in B \text{ or } A^c \in B\}$. As in the Kripke case, $[\![\Delta]\!]$ is not separating for $\mathcal{N}$.

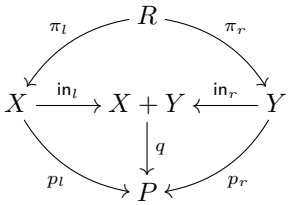
▶ **Example 7.** The language of *instantial neighbourhood logic (INL)* [25] arises from the similarity type $\Lambda = \{\Box_n \mid n \in \mathbb{N}\}$ where $\Box_n$ is $n+1$-ary for all $n \in \mathbb{N}$. The semantics of instantial neighbourhood logic is obtained by taking $T = \mathcal{P}\mathcal{P}$ and $[\![\Box_n]\!]_X(A_1, ..., A_n, B) = \{N \in \mathcal{P}\mathcal{P}X \mid \exists U \in N : U \subseteq B \text{ and for all } i = 1, \ldots, n : \ U \cap A_i \neq \emptyset\}$. The collection $\{[\![\Box_n]\!] \mid n \in \mathbb{N}\}$ is separating for $\mathcal{P}\mathcal{P}_\omega$, where $\mathcal{P}_\omega(X)$ are all finite subsets of X: Suppose $N, N' \in \mathcal{P}(\mathcal{P}_\omega(X))$ and $B \in N \setminus N'$ with $B = \{x_1, \ldots, x_n\}$. Then $[\![\Box_n]\!](\{x_1\}, \ldots, \{x_n\}, B)$ contains N, but not N'. It is not hard to see that any finite subset of $\{[\![\Box_n]\!] \mid n \in \mathbb{N}\}$ is not separating for $\mathcal{P}\mathcal{P}_\omega$.

2.2 Relations and Coherence

Let $R \subseteq X \times Y$ be a relation. The converse of R is written $R^{-1} \subseteq Y \times X$. The R-image of $U \subseteq X$ is the set $R[U] = \{y \in Y \mid \exists x \in U : (x, y) \in R\}$. If $R \subseteq X \times X$ is an equivalence relation, then we write $[x]_R$ (or simply $[x]$) for the equivalence class of x. The composition of $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ is $R; S \subseteq X \times Z$.

We will use pullbacks and pushouts in what follows. We recall the concrete constructions in Sets, and refer to [17] for the general definitions. In Sets, a pullback $(B, g_l\colon B \to X, g_r\colon B \to Y)$ of two functions $f_l\colon X \to Z$ and $f_r\colon Y \to Z$ can be concretely constructed by taking $B = pb(f_l, f_r) = \{(x, y) \in X \times Y \mid f_l(x) = f_r(y)\}$ and $g_l = \pi'_l\colon B \to X$ and $g_r = \pi'_r\colon B \to Y$ to be the projections. Pushouts are the dual notion of pullbacks. Given a relation $R \subseteq X \times Y$ the pushout of the projections $\pi_l\colon R \to X$ and $\pi_r\colon R \to Y$, is obtained concretely as follows.

The relation R can be seen as a relation R_{X+Y} on the coproduct $X + Y$ by composing the projections with the coproduct injections $\mathsf{in}_l\colon X \to X + Y$ and $\mathsf{in}_r\colon Y \to X + Y$. More precisely, $R_{X+Y} = (\mathsf{in}_l \times \mathsf{in}_r)(R) = \{(\mathsf{in}_l(x), \mathsf{in}_r(y)) \mid (x, y) \in R\}$. Let $\overline{R}$ be the smallest equivalence relation on $X + Y$ that contains R_{X+Y}. Then we take $P = (X + Y)/\overline{R}$ to be the set of $\overline{R}$-equivalence classes with associated quotient map $q\colon X + Y \to P$, and we take $p_l = q \circ \mathsf{in}_l\colon X \to P$, $p_r = q \circ \mathsf{in}_r\colon Y \to P$. Then (P, p_l, p_r) is a pushout of π_l and π_r. The situation is illustrated with the diagram below.



Our definition of Λ-bisimulation relies on the notion of coherent pairs, which was introduced in [8]. We recall the definition and some basic facts.

▶ **Definition 8** (R-coherent pairs)**.** Let $R \subseteq X \times Y$ be a relation with projections $\pi_l\colon R \to X$ and $\pi_r\colon R \to Y$, and let $U \subseteq X$ and $V \subseteq Y$. The pair (U, V) is *R-coherent* if $R[U] \subseteq V$ and $R^{-1}[V] \subseteq U$. In case $R \subseteq X \times X$ and $U \subseteq X$, then we say that U is *R-coherent* if (U, U) is R-coherent.

Note that if R is an equivalence relation on a set X and $U \subseteq X$, then U is R-coherent iff U is R-closed, i.e., U is a union of R-equivalence classes. We make some easy, but useful observations. Further properties of coherent sets may be found in Lemma 2.2 and 2.3 of [8].

▶ **Lemma 9.**

1. *Let $R \subseteq X \times Y$ be a relation with projections $\pi_l \colon R \to X$ and $\pi_r \colon R \to Y$, and let $U \subseteq X$ and $V \subseteq Y$. The following are equivalent:*
   - **(a)** *(U, V) is R-coherent.*
   - **(b)** *$\pi_l^{-1}[U] = \pi_r^{-1}[V]$.*
   - **(c)** *(U, V) is in the pullback of $Q\pi_l$ and $Q\pi_r$.*
   - **(d)** *for all $(x, y) \in R$, $x \in U$ iff $y \in V$.*
   - **(e)** *$U + V$ is R_{X+Y}-coherent.*
2. *If $R \subseteq X \times X$ is reflexive and (U, V) is R-coherent, then $U = V$.*

Due to Lemma 9(1.c), we will refer to the concrete pullback $(pb(Q\pi_l, Q\pi_r), \pi_l', \pi_r')$ as the *pullback of R-coherent pairs*.

The following lemma shows that there is a fundamental connection between coherent pairs and pushouts of relations. It is also key in proving Propositions 20 and 21 later.

▶ **Lemma 10.** *Let $R \subseteq X \times Y$ be a relation, and let (P, p_l, p_r) be the pushout of R. The triple (QP, Qp_l, Qp_r) is also pullback of $(QR, Q\pi_l, Q\pi_r)$, and hence it is isomorphic to $(pb(Q\pi_l, Q\pi_r), \pi_l', \pi_r')$, the pullback of R-coherent pairs.*

Proof. This lemma holds for the general reason that the contravariant powerset functor $Q \colon \mathsf{Sets} \to \mathsf{Sets}^{\mathrm{op}}$ is a left adjoint of itself, more precisely of $Q^{\mathrm{op}} \colon \mathsf{Sets}^{\mathrm{op}} \to \mathsf{Sets}$, and that left adjoints preserve colimts. Hence Q turns the pushout into a pullback. Since pullbacks are unique up to isomorphism, the result follows. The isomorphism is given concretely by the map $h \colon QP \to pb(Q\pi_l, Q\pi_r)$ defined for all $A \in QP$ by $h(A) = (Qp_l(A), Qp_r(A))$. We verify that $(Qp_l(A), Qp_r(A))$ is R-coherent. So let $(x, y) \in R$. It follows that $p_l(x) = p_r(y)$, and hence $x \in Qp_l(A)$ iff $p_l(x) \in A$ iff $p_r(y) \in A$ iff $y \in Qp_r(A)$. To see that h is injective, suppose $A, A' \subseteq P$ and $a \in A \setminus A'$. The maps p_l and p_r are jointly surjective. If $a \in p_l[X]$, then there is a $x \in Qp_l(A)$ such that $p_l(x) = a$. If also $x \in Qp_l(A')$, then $p_l(x) = a \in A'$, a contradiction. Similarly, if $a \in p_r[Y]$, then there is a $y \in Qp_r(A)$ such that $p_r(y) = a$, and it must be the case that $y \notin Qp_r(A')$. Hence $h(A) \neq h(A')$. To see why h is surjective, it can be verified that if (U, V) is R-coherent, and we take $A \subseteq P$ to be $A = p_l[U] \cup p_r[V]$, then $h(A) = (U, V)$. For example, to see why $Qp_l(p_l[U]) = U$, first note that the inclusion $\supseteq$ always holds. Equality follows from the fact that (U, V) is R-coherent. Finally, we remark that (QP, Qp_l, Qp_r) is a competitor to the pullback of R-coherent pairs precisely because $(Qp_l(A), Qp_r(A))$ is R-coherent for all $A \subseteq P$. ◀

3 Λ-bisimulation

In this section, we introduce the notion of Λ-bisimulation between T-coalgebras, and investigate its properties. This notion is parametric in the choice of a signature Λ and a Λ-structure $(T, (\llbracket \heartsuit \rrbracket)_{\heartsuit \in \Lambda})$. In the remaining part of the paper, we therefore assume that we have fixed a functor $T \colon \mathsf{Sets} \to \mathsf{Sets}$, and for each $\heartsuit \in \Lambda$, a predicate lifting $\llbracket \heartsuit \rrbracket$ of appropriate arity. From now on, by abuse of language, we will also refer to Λ as the set of these predicate liftings. Moreover, we let $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ denote arbitrary T-coalgebras.

3.1 Definition and Basic Properties

Our definition of Λ-bisimulations is as follows.

▶ **Definition 11** (Λ-bisimulation). Let $Z \subseteq X \times Y$ be a relation and let $(pb(Q\pi_l, Q\pi_r), \overline{\pi}_l, \overline{\pi}_r)$ be the associated pullback of Z-coherent pairs. The relation Z is a Λ-*bisimulation* between $\mathbb{X}$ and $\mathbb{Y}$, if for all $\heartsuit \in \Lambda$, with $\heartsuit$ n-ary:

$$Q\pi_l \circ Q\gamma \circ [\![\heartsuit]\!]_X \circ \overline{\pi}_l^n = Q\pi_r \circ Q\delta \circ [\![\heartsuit]\!]_Y \circ \overline{\pi}_r^n \tag{1}$$

where $\overline{\pi}_l^n \colon pb(Q\pi_l, Q\pi_r)^n \to (QX)^n$ and $\overline{\pi}_r^n \colon pb(Q\pi_l, Q\pi_r)^n \to (QY)^n$ are the pointwise projections, for example, $\overline{\pi}_l((U_1, V_1), \ldots, (U_n, V_n)) = (U_1, \ldots, U_n)$. In other words, the relation Z is a Λ-bisimulation if whenever $(x, y) \in Z$, then for all $\heartsuit \in \Lambda$, n-ary, and all Z-coherent pairs $(U_1, V_1), \ldots, (U_n, V_n)$, we have that

$$\gamma(x) \in [\![\heartsuit]\!]_X(U_1, \ldots, U_n) \quad \text{iff} \quad \delta(y) \in [\![\heartsuit]\!]_Y(V_1, \ldots, V_n). \tag{\textbf{Coherence}}$$

We write $\mathbb{X}, x \sim_\Lambda \mathbb{Y}, y$, if there is a Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$ that contains (x, y). A Λ-bisimulation on a T-coalgebra $\mathbb{X}$ is a Λ-bisimulation between $\mathbb{X}$ and $\mathbb{X}$.

The next lemma provides an easy observation about dual modal operators that we will use further in the examples.

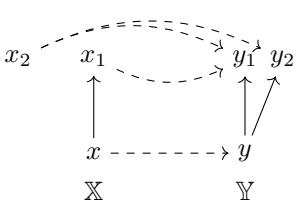
▶ **Lemma 12.** *Let $\heartsuit, \heartsuit' \in \Lambda$ be two n-ary dual modalities, that is $\heartsuit = \neg\heartsuit'\neg$. A relation Z is a $\heartsuit$-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$ iff Z is a $\heartsuit'$-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$.*

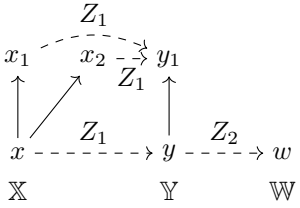
Proof. First, $\heartsuit = \neg\heartsuit'\neg$ means that for all sets W, and all $A_1, \ldots, A_n \subseteq W$, we have that $[\![\heartsuit]\!]_W(A_1, \ldots, A_n) = ([\![\heartsuit']\!]_W(A_1^c, \ldots, A_n^c))^c$. We note that if $Z \subseteq X \times Y$, $U \subseteq X$ and $V \subseteq Y$, then the pair (U, V) is Z-coherent iff (U^c, V^c) is Z-coherent. Hence, $\gamma(x) \in [\![\heartsuit]\!]_X(U_1, ..., U_n)$ iff $\gamma(x) \notin [\![\heartsuit']\!]_X(U_1^c, ..., U_n^c)$ iff $\delta(y) \notin [\![\heartsuit']\!]_Y(V_1^c, ..., V_n^c)$ iff $\delta(y) \in [\![\heartsuit]\!]_Y(V_1, ..., V_n)$. ◀

We provide some examples of our notion of Λ-bisimulation.

▶ **Example 13.** Taking $T = \mathcal{P}$ (i.e. T-coalgebras are Kripke frames) and $\Lambda = \{\Box\}$ (or $\Lambda = \{\Diamond\}$), then a relation Z between Kripke frames $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ is a $\Box$-bisimulation if for all $(x, y) \in Z$ and all Z-coherent pairs (U, V): $\gamma(x) \subseteq U$ iff $\delta(y) \subseteq V$. An easy proof shows that if Z is a Kripke bisimulation then Z is a $\Box$-bisimulation. However, a Λ-bisimulation may not be a Kripke bisimulation. Consider the following Kripke frames: $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$, where $X = \{x, x_1, x_2\}$, $\gamma(x) = \{x_1\}$, $Y = \{y, y_1, y_2\}$ and $\delta(y) = \{y_1, y_2\}$. It can be easily checked that the relation $Z = \{(x, y), (x_1, y_1), (x_2, y_1), (x_2, y_2)\}$ is a $\Box$-bisimulation, but it is not a Kripke bisimulation, since the successor y_2 of y is not related to a successor of x. The situation is depicted below on the left, where Z is indicated by dashed lines. Still, when considering the associated bisimilarity notions, we find that Λ-bisimilarity coincides with Kripke bisimilarity. This follows from our Proposition 22, using that $\Box$ (and $\Diamond$) is separating and $\mathcal{P}$ preserves weak pullbacks.

This choice of T and Λ demonstrates that, in general, Λ-bisimulations are not closed under relational composition. To see this, let $\mathbb{X} = (X, \gamma)$, $\mathbb{Y} = (Y, \delta)$ and $\mathbb{W} = (W, \alpha)$ be the three Kripke frames depicted below on the right together with the two relations $Z_1 \subseteq X \times Y$ and $Z_2 \subseteq Y \times W$ (indicated by dashed lines): It is straightforward to check that Z_1 and Z_2 are Λ-bisimulations, but the composition $Z_1; Z_2 = \{(x, w)\}$ is not, because $(\{x, x_1\}, \{w\})$ is $Z_1; Z_2$-coherent and $\gamma(x) \not\subseteq \{x, x_1\}$ and $\gamma(x) \not\subseteq \{x_2\}$, whereas $\alpha(w) \subseteq \{w\}$.





▶ **Example 14.** Taking $T = \mathcal{N}$ (i.e. neighbourhood frames) and $\Lambda = \{\Box\}$, where $\Box$ is the neighbourhood modality from Example 5, we find that a relation Z is a $\Box$-bisimulation between neighbourhood frames $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ if for all $(x, y) \in Z$ and all Z-coherent (U, V): $U \in \gamma(x)$ iff $V \in \delta(y)$. This shows that Λ-bisimulations are the same as *precocongruences* which were introduced in [8], due to [8, Proposition 3.16]. We will discuss the relation between precocongruences and Λ-bisimulations further in subsection 3.2.

▶ **Example 15.** Taking $T = \mathcal{P}$ and $\Lambda = \{\Delta\}$, where Δ is the contingency modality from Example 4, then a Z is Δ-bisimulation between Kripke frames $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ if for all $(x, y) \in Z$ and all Z-coherent (U, V): $\gamma(x) \subseteq U$ *or* $\gamma(x) \subseteq U^c$ iff $\delta(y) \subseteq V$ *or* $\delta(y) \subseteq V^c$. This is exactly the definition of a *rel-Δ-bisimulation* which was introduced in [2]. Prop. 3.4 in [6] tells us that Δ-bisimilarity does not imply $\Box$-bisimilarity. Note that in [2] the relation $\sim_\Lambda$ is denoted $\sim_\Delta^{\mathsf{betw}}$.

▶ **Example 16.** Taking $T = \mathcal{N}$ and $\Lambda = \{\Delta\}$, where Δ is the neighbourhood contingency modality from Example 6, then by instantiating (**Coherence**) for Δ, we have that Z is a Δ-bisimulation between neighbourhood frames $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ if for all $(x, y) \in Z$ and all Z-coherent (U, V): $U \in \gamma(x)$ *or* $U^c \in \gamma(x)$ iff $V \in \delta(y)$ *or* $V^c \in \delta(y)$. This is exactly the definition of a *nbh-Δ-bisimulation* which was introduced in [2].

The following proposition shows that Λ-bisimulations enjoy many of the properties known to hold for Kripke bisimulations. In particular, even though Λ-bisimulations do not need to be closed under composition (cf. Example 13), we can still show that on a single T-coalgebra, $\sim_\Lambda$ is an equivalence relation.

▶ **Proposition 17.** *Let $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ be T-coalgebras.*

1. *The identity relation* $\mathrm{Id} \subseteq X \times X$ *is a Λ-bisimulation on $\mathbb{X}$.*
2. *If $Z \subseteq X \times Y$ is a Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$ then $Z^{-1} \subseteq Y \times X$ is a Λ-bisimulation between $\mathbb{Y}$ and $\mathbb{X}$.*
3. *Λ-bisimulations are closed under arbitrary unions: If $Z_i \subseteq X \times Y$, $i \in I$, are Λ-bisimulations, then so is $\bigcup_{i \in I} Z_i$.*
4. *The relation $\sim_\Lambda$ is the largest Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$.*
5. *The relation $\sim_\Lambda$ on $\mathbb{X}$ is an equivalence relation.*

Proof.

Item 1-2: are straightforward to check. We omit the details.

Item 3: Let $Z_i \subseteq X \times Y$, $i \in I$, be Λ-bisimulations, and let $Z = \bigcup_{i \in I} Z_i$. To show that Z is a Λ-bisimulation, assume that $(x, y) \in Z$, $\heartsuit \in \Lambda$, and (U, V) is a Z-coherent pair. From $(x, y) \in Z$ it follows that $(x, y) \in Z_i$ for some $i \in I$, and since $Z_i \subseteq Z$ we also have that (U, V) is Z_i-coherent. Hence $\gamma(x) \in [\![\heartsuit]\!]_X(U) \iff \delta(y) \in [\![\heartsuit]\!]_X(V)$. Which implies that Z is a Λ-bisimulation.

Item 4: Follows immediately from item 3.

Item 5: We show that if Z is a Λ-bisimulation on $\mathbb{X}$, then the equivalence closure of Z is again a Λ-bisimulation on $\mathbb{X}$, which suffices due to item 4. So let Z be a Λ-bisimulation on $\mathbb{X}$. By items 1 and 2, we may assume that Z is reflexive and symmetric. The result follows by showing that the transitive closure $Z^+ = \bigcup_{n \geq 1} Z^n$ is a Λ-bisimulation. Due to item 3 it suffices to show that for all $n \geq 1$, Z^n is a Δ-bisimulation. The proof is by induction on n. The base case ($n = 1$) holds by assumption on Z. Assume it holds for n. Induction step ($n + 1$): First note that if (U, U'') is Z^{n+1}-coherent, then since Z^{n+1} is reflexive, it follows that $U = U''$. Now suppose $(x, x') \in Z^n$, $(x', x'') \in Z$ and (U, U) is Z^{n+1}-coherent. Since Z and Z^n are reflexive and $Z^{n+1} = Z^n; Z$, it follows that

$Z \subseteq Z^{n+1}$ and $Z^n \subseteq Z^{n+1}$, and hence (U, U) is Z-coherent as well as Z^n-coherent. We then have

$$\begin{array}{rcll} \gamma(x) \in [\![\heartsuit]\!]_X(U) & \iff & \gamma(x') \in [\![\heartsuit]\!]_X(U) & \text{(by induction hypothesis)} \\ & \iff & \gamma(x'') \in [\![\heartsuit]\!]_X(U) & \text{(since } Z \text{ is a } \Lambda\text{-bisimulation).} \end{array}$$

Hence Z^{n+1} is a Λ-bisimulation which concludes the proof. ◀

Λ-bisimulations were designed to match the expressiveness of the modal language. In the next proposition we show that indeed, Λ-bisimilar states satisfy the same $\mathcal{L}_\Lambda$-formulas.

▶ **Proposition 18.** *If* $\mathbb{X}, x \sim_\Lambda \mathbb{Y}, y$ *then* $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$.

Proof. Let $\mathbb{X}, x \sim_\Lambda \mathbb{Y}, y$, so there exists a Λ-bisimulation $Z \subseteq X \times Y$ such that $(x, y) \in Z$. The proof is by induction on φ. The only interesting part is the modal case of the inductive step. Assume that φ is of the form $\heartsuit\psi$. By induction hypothesis, $([\![\psi]\!]_{\mathbb{X}}, [\![\psi]\!]_{\mathbb{Y}})$ is Z-coherent. Since Z is a Λ-bisimulation, we have $\gamma(x) \in [\![\heartsuit]\!]_X([\![\psi]\!]_{\mathbb{X}})$ iff $\delta(y) \in [\![\heartsuit]\!]_Y([\![\psi]\!]_{\mathbb{Y}})$, which means that $\mathbb{X}, x \models \heartsuit\psi$ iff $\mathbb{Y}, y \models \heartsuit\psi$. ◀

3.2 Comparison with other notions

In this part, we compare our notion of Λ-bisimulation to the established notions of T-bisimulations and behavioural equivalence. It turns out that Λ-bisimulations are closest to the notion called *precocongruences* in [8]. Finally, we also compare our notion to other similar proposals by Gorín and Schröder [7].

First we recall the definitions of behavioural equivalence, T-bisimulations and precocongruences.

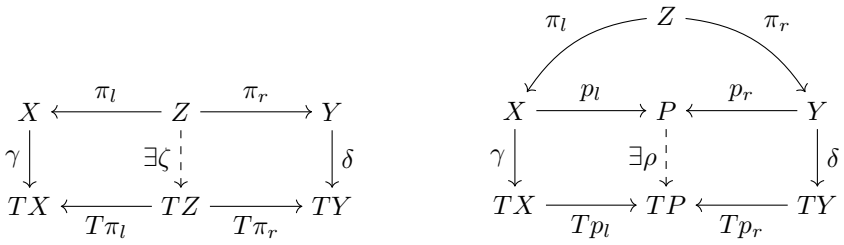
▶ **Definition 19.** Let $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ be T-coalgebras.

- *Behavioural equivalence.* Two states $x \in X$ and $y \in Y$ are *behaviourally equivalent* (notation: $\mathbb{X}, x \sim_{bh} \mathbb{Y}, y$), if there is a T-coalgebra $\mathbb{E} = (E, \epsilon)$ and a pair of T-coalgebra morphisms $f : \mathbb{X} \to \mathbb{E}$ and $g : \mathbb{Y} \to \mathbb{E}$ such that $f(x) = g(y)$.
- *T-bisimulation.* A relation $Z \subseteq X \times Y$ is a *T-bisimulation* between X and Y, if there exists a function $\zeta : Z \to TZ$ such that the projections $\pi_l : Z \to X$ and $\pi_r : Z \to Y$ are T-coalgebra morphisms, i.e., the diagram in Figure 1a commutes. Two states $x \in X$ and $y \in Y$ are T-bisimilar (notation: $\mathbb{X}, x \sim_T \mathbb{Y}, y$) if there is a T-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$ linking x and y.
- *Precocongruence.* Let $Z \subseteq X \times Y$ be a relation with pushout (P, p_l, p_r). Z is a *precocongruence* between $\mathbb{X}$ and $\mathbb{Y}$ if there exists a function $\rho : P \to TP$ such that the pushout morphisms $p_l : X \to P$ and $p_r : Y \to P$ are T-coalgebra morphisms, i.e., if the diagram in Figure 1b commutes. If two states $x \in X$ and $y \in Y$ are related by some precocongruence, we write $\mathbb{X}, x \sim_p \mathbb{Y}, y$.

In the following proposition we give the first comparison between precocongruences, T-bisimulations and Λ-bisimulations.

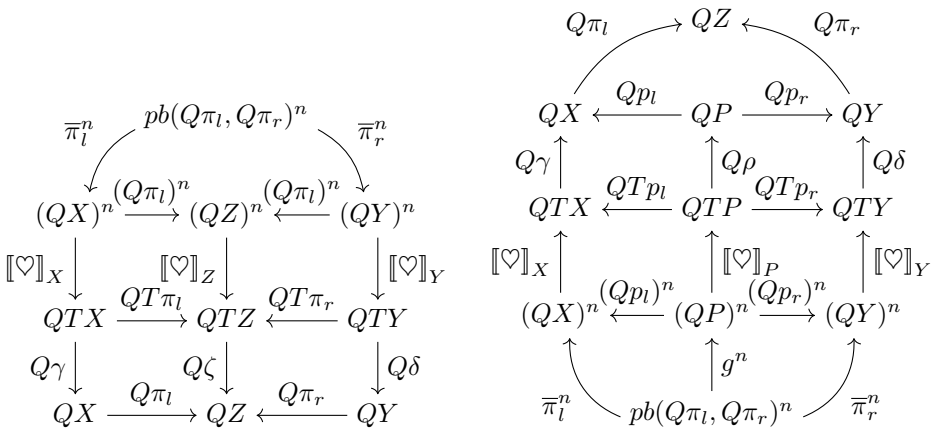
▶ **Proposition 20.** *Let* $\mathbb{X} = (X, \gamma)$ *and* $\mathbb{Y} = (Y, \delta)$ *be T-coalgebras, and Z be a relation between X and Y.*

1. *If Z is a T-bisimulation then Z is a Λ-bisimulation.*
2. *If Z is a precocongruence then Z is a Λ-bisimulation.*



(a) Z is a T-bisimulation. **(b)** Z is a precocongruence.

Figure 1



(a) T-bisimulations are Λ-bisimulations. **(b)** Precocongruences are Λ-bisimulations.

Figure 2

Proof.

Item 1: Apply Q to the diagram of T-bisimulation (Figure 1a), and take the pullback of $Q\pi_l$ and $Q\pi_r$. Then, by naturality of $[\![\heartsuit]\!]$, and the fact that π_l and π_r are coalgebra morphisms, the diagram in Figure 2a commutes and hence, Z is a Λ-bisimulation.

Item 2: Let $Z \subseteq X \times Y$ be a precocongruence relation with pushout (P, p_l, p_r), and let (U, V) be Z-coherent. By Lemma 10, there is a map $g\colon pb(Q\pi_l, Q\pi_r) \to QP$ such that $Qp_l \circ g = \overline{\pi}_l$ and $Qp_r \circ g = \overline{\pi}_r$. Then, by naturality of $[\![\heartsuit]\!]$ and the fact that p_l and p_r are T-coalgebra morphisms, it follows that the outer part of the diagram in Figure 2b commutes. Hence, Z is a Λ-bisimulation. ◀

The next proposition shows that, if Λ is separating, then we have the converse of Proposition 20(2).

▶ **Proposition 21.** *If Λ is separating and $Z \subseteq X \times Y$ is a Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$, then Z is a precocongruence between $\mathbb{X}$ and $\mathbb{Y}$.*

Proof. Let $Z \subseteq X \times Y$ be a Λ-bisimulation with projections $\pi_l : Z \to X$ and $\pi_r : Z \to Y$, and pushout (P, p_l, p_r) We need to define $\rho : P \to TP$ such that $\rho \circ p_l = Tp_l \circ \gamma$ and $\rho \circ p_r = Tp_r \circ \delta$. We obtain such a ρ from the universal property of the pushout, if we can show that for all $(x, y) \in Z$: $Tp_l(\gamma(x)) = Tp_r(\delta(y))$. To prove this, since Λ is separating, it suffices to show that for arbitrary $\heartsuit \in \Lambda$, n-ary, and $A_1, \ldots, A_n \subseteq P$, $Tp_l(\gamma(x)) \in [\![\heartsuit]\!]_P(A_1, \ldots, A_n)$ iff $Tp_r(\delta(y)) \in [\![\heartsuit]\!]_P(A_1, \ldots, A_n)$, which is equivalent to, $Q\pi_l \circ Q\gamma \circ QTp_l \circ [\![\heartsuit]\!]_P = Q\pi_r \circ Q\delta \circ QTp_r \circ [\![\heartsuit]\!]_P$. This holds because of the commutativity of the diagram in Figure 21, where the map h is obtained from Lemma 10. ◀

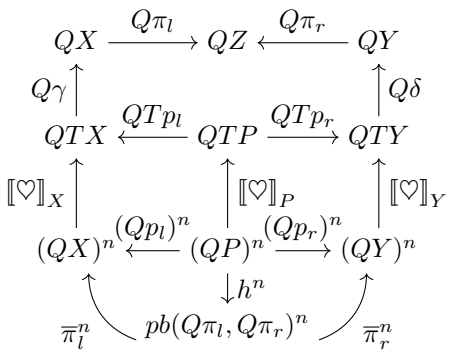


Figure 3 Proof of Proposition 21.

It was shown in [8, Proposition 3.10] that, in general, T-bisimilarity implies precocongruence equivalence which in turn implies behavioural equivalence. This fact together with Proposition 21 tells us that Λ-bisimilarity implies behavioural equivalence, whenever Λ is separating. Moreover, it is well known [23] that if T preserves weak pulbacks, then T-bisimilarity coincides with behavioural equivalence. Hence in this case, by Proposition 21, it follows that Λ-bisimilarity coincides with T-bisimilarity and behavioural equivalence. The following proposition summarises our discussion so far.

▶ **Proposition 22.** *Let Λ be a set of predicate liftings for T.*

1. $\mathbb{X}, x \sim_T \mathbb{Y}, y \implies \mathbb{X}, x \sim_p \mathbb{Y}, y \implies \mathbb{X}, x \sim_\Lambda \mathbb{Y}, y.$
2. *If Λ is separating, then*
$$\mathbb{X}, x \sim_p \mathbb{Y}, y \iff \mathbb{X}, x \sim_\Lambda \mathbb{Y}, y \implies \mathbb{X}, x \sim_{bh} \mathbb{Y}, y.$$
3. *If Λ is separating and T preserves weak pullbacks, then all four notions coincide:*
$$\mathbb{X}, x \sim_T \mathbb{Y}, y \iff \mathbb{X}, x \sim_p \mathbb{Y}, y \iff \mathbb{X}, x \sim_\Lambda \mathbb{Y}, y \iff \mathbb{X}, x \sim_{bh} \mathbb{Y}, y.$$

The next lemma states that similar to the fact that T-coalgebra morphisms preserve and reflect behavioural equivalence, one can show that they preserve and reflect Λ-bisimilarity as well. We will use this fact to prove the Hennessy-Milner theorem in Section 4.

▶ **Proposition 23.** *If $f : \mathbb{X} \to \mathbb{Y}$ is a T-coalgebra morphism, then for all $x, x' \in X$:*

$$\mathbb{X}, x \sim_\Lambda \mathbb{X}, x' \quad \textit{iff} \quad \mathbb{Y}, f(x) \sim_\Lambda \mathbb{Y}, f(x').$$

Proof. For the direction from left to right, assume $\mathbb{X}, x \sim_\Lambda \mathbb{X}, x'$. Then, there exists a Λ-bisimulation Z on $\mathbb{X}$ such that $(x, x') \in Z$. We show that $(f \times f)(Z) = \{(f(x), f(x')) \in Y \times Y \mid (x, x') \in Z\}$ is a Λ-bisimulation. Let $(f(x), f(x')) \in (f \times f)(Z)$ and $\heartsuit \in \Lambda$. Note that if (U, V) is $(f \times f)(Z)$-coherent, then the pair $(f^{-1}[U], f^{-1}[V])$ is Z-coherent. By naturality and the fact that f is a coalgebra morphism, we have $\delta(f(x)) \in [\![\heartsuit]\!]_Y(U)$ iff $\gamma(x) \in [\![\heartsuit]\!]_X(f^{-1}[U])$, and $\delta(f(x')) \in [\![\heartsuit]\!]_Y(V)$ iff $\gamma(x') \in [\![\heartsuit]\!]_X(f^{-1}[V])$. Since Z is a Λ-bisimulation and $(f^{-1}[U], f^{-1}[V])$ is Z-coherent, we obtain $\delta(f(x)) \in [\![\heartsuit]\!]_Y(U)$ iff $\delta(f(x')) \in [\![\heartsuit]\!]_Y[V]$. A similar argument shows that if Z is a Λ-bisimulation on $\mathbb{Y}$ then $(f^{-1} \times f^{-1})(Z) = \{(x, x') \in X \times X \mid (f(x), f(x')) \in Z\}$ is a Λ-bisimulation on X. ◀

3.2.1 Λ-bisimulations: a different approach

Gorín and Schröder introduced in [7] a similar notion of Λ-bisimulation. To distinguish their notion from the one presented here, we refer to it as GS-Λ-bisimulation. One difference with

our work is that Gorín and Schröder assume that Λ is a set of *monotone* predicate liftings. For convenience, we recall their definition here, which can be stated without the assumption of monotonicity. A relation $Z \subseteq X \times Y$ is a *GS-Λ-bisimulation* if whenever $(x, y) \in Z$ then for all $\heartsuit \in \Lambda$ and for all $A \subseteq X$ and $B \subseteq Y$

$$\gamma(x) \in [\![\heartsuit]\!]_X(A) \Rightarrow \delta(y) \in [\![\heartsuit]\!]_Y(Z[A]) \quad \text{and} \quad \delta(y) \in [\![\heartsuit]\!]_Y(B) \Rightarrow \gamma(x) \in [\![\heartsuit]\!]_X(Z^{-1}[B]).$$

Under the assumption that all $\heartsuit \in \Lambda$ are monotone, it is straightforward to show that a GS-Λ-bisimulation is also Λ-bisimulation. Example 13 demonstrates that there exists a choice of T and monotone Λ such that the two notions differs at the level of relations. Namely, the relation Z given there is a Λ-bisimulation, but not a GS-Λ-bisimulation. To see this, take $A = \{x_1, x_2\}$. We have that $\gamma(x) = \{x_1\} \subseteq A$, but $\delta(y) = \{y_1, y_2\} \not\subseteq Z[A] = \{y_1\}$. However, one can show that under the assumption that Λ is monotone, difunctional (also called zig-zag closed) Λ-bisimulations are GS-Λ-bisimulations, and that the relation $\sim_\Lambda$ between any two T-coalgebras is difunctional. Hence the two bisimilarity notions coincide. In [7, Theorem 26] it was shown that when Λ is separating and monotone, then GS-Λ-bisimilarity coincides with behavioural equivalence, and hence under these assumptions, Λ-bisimilarity coincides both with GS-Λ-bisimilarity and with behavioural equivalence.

▶ **Proposition 24.** *If Λ is separating and monotone, then*

$$\mathbb{X}, x \sim_{GS\text{-}\Lambda} \mathbb{Y}, y \quad \Longleftrightarrow \quad \mathbb{X}, x \sim_\Lambda \mathbb{Y}, y \quad \Longleftrightarrow \quad \mathbb{X}, x \sim_{bh} \mathbb{Y}, y.$$

We point out that our results on Λ-bisimulation do not require Λ to be monotone. Furthermore, our aims and results differ from those of [7] where the starting point was to investigate simulations between T-coalgebras. In this context, GS-Λ-bisimulations arose naturally as two-way simulations. The results in [7] focus on identifying conditions that ensure that GS-Λ-bisimilarity coincides with behavioural equivalence and/or T-bisimilarity. Our approach is to accept that the language is not expressive, and show that Λ-bisimilarity allows us to generalise several results that are known to hold for expressive languages.

▶ **Example 25.** Consider INL from Example 7 (i.e. $T = \mathcal{PP}$). Since $[\![\Box_n]\!]$ is monotone [25], it follows that $\Box_n$-bisimilarity coincides with GS-$\Box_n$-bisimilarity. Note that $[\![\Box_0]\!]_X(A) = \{N \in \mathcal{PP}(X) \mid \exists U \in N : U \subseteq A\}$ is like the monotone neighbourhood modality (which is usually interpreted in $\mathcal{N}$-coalgebras). It is straightforward to prove that GS-$\Box_0$-bisimulations coincide with monotonic bisimulations (see e.g. [25]). For $n \geq 1$, one can show that Z is a GS-$\Box_n$-bisimulation iff for all $(x, y) \in Z$: (Here $A \subseteq_n B$ means that $A \subseteq B$ and $|A| \leq n$.)
(forth)$_n$ $\forall U \neq \emptyset \colon U \in \gamma(x) \Longrightarrow \forall U' \subseteq_n U. \exists V \neq \emptyset \colon V \in \delta(y), V \subseteq Z[U]$ and $U' \subseteq Z^{-1}[V]$.
(back)$_n$ $\forall V \neq \emptyset \colon V \in \delta(y) \Longrightarrow \forall V' \subseteq_n V. \exists U \neq \emptyset \colon U \in \gamma(x), U \subseteq Z^{-1}[V]$ and $V' \subseteq Z[U]$.
The proof uses the fact that if $\{x_1, \ldots, x_n\} \subseteq A \in N$ then $N \in \Box_n(\{x_1\}, \ldots, \{x_n\}, A)$.

3.3 Λ-morphisms

Given the fact that the graph of a T-coalgebra morphism is a T-bisimulation (cf. [23, Theorem 2.5.]), it is it is natural to define a Λ-*morphism from* $\mathbb{X}$ *to* $\mathbb{Y}$ to be a a function $f \colon X \to Y$ for which the graph $Gr(f) = \{(x, f(x)) \mid x \in X\}$ is a Λ-bisimulation. It then follows from Proposition 20(1) that T-coalgebra morphisms are also Λ-morphisms. Moreover, one can show that Λ-homomorphism are closed under composition (unlike Λ-bisimulations). Therefore, T-coalgebras together with Λ-morphisms form a category.

In Enqvist [4], a weak notion of morphism for T-coalgebras was proposed which, like ours, is parametric in a set Λ of predicate liftings. To distinguish this notion from ours, we refer

to it as E-Λ-morphism. We briefly recall the definition (which we state only for unary $\heartsuit$, as is the case in [4]). A function $f\colon X \to Y$ is an *E-Λ-morphism from $\mathbb{X}$ to $\mathbb{Y}$* if for all $B \subseteq Y$, $x \in X$, and $\heartsuit \in \Lambda$: $\delta(f(x)) \in [\![\heartsuit]\!]_Y(B)$ implies $\gamma(x) \in [\![\heartsuit]\!]_X(f^{-1}[B])$. Taking $Z = Gr(f)$, it can easily be seen that a pair (U, V) is Z-coherent iff $U = f^{-1}[V] = Q(V)$. It then follows that Λ-morphisms are E-Λ-morphisms. Since only one direction of the **(Coherence)** condition needs to hold for E-Λ-morphisms, it is straightforward to construct an example of a E-Λ-morphism which is not a Λ-morphism.

We do not investigate our notion of Λ-morphisms further in the present paper. Several interesting questions could be asked, though. We discuss those in Section 5.

4 Hennessy-Milner Theorem

This section is devoted to proving the main technical result of the paper: a coalgebraic Hennessy-Milner theorem for our notion of Λ-bisimilarity.

As we saw in Proposition 18, $\mathcal{L}_\Lambda$-formulas are invariant under Λ-bisimulation. Given that our modal language has only finite conjunctions, we will need to assume our coalgebra functor is finitary. This is the analogue of restricting to image-finite Kripke frames, as is done in the classic Hennessy-Milner theorem. However, there is another issue. As shown in [2, Example 1(4)], even between finite $\mathcal{P}$-coalgebras, it is possible for two states to fail to be Λ-bisimilar while still satisfying the same modal $\mathcal{L}_\Lambda$-formulas. We recall this example here for convenience. We are in the setting of contingency logic over Kripke frames from Example 4, i.e. $\Lambda = \{\Delta\}$. Let $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ be two $\mathcal{P}$-coalgebras, where $X = \{x, x_1, x_2\}$, $\gamma(x) = \{x_1, x_2\}$, $\gamma(x_i) = \emptyset$ for $i = 1, 2$, $Y = \{y\}$ and $\delta(y) = \emptyset$. The relation $Z = \{(x, y), (x_1, x_2), (x_2, x_2)\}$ is a Λ-bisimulation on the coproduct of $\mathbb{X}$ and $\mathbb{Y}$ (we omit injection maps for readability). Since the coproduct injections are T-coalgebra morphisms, they are also Λ-morphisms, and hence $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$. However, it is not hard to show that there is no Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$ linking x and y. The solution in [2] was to define a notion of bisimilarity via the coproduct of Kripke/neighbourhood frames. We take a similar approach here.

▶ **Definition 26.** Two states x in $\mathbb{X}$ and y in $\mathbb{Y}$ are Λ_+-bisimilar (notation: $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$) if $\mathbb{X} + \mathbb{Y}, \mathsf{in}_l(x) \sim_\Lambda \mathbb{X} + \mathbb{Y}, \mathsf{in}_r(y)$.

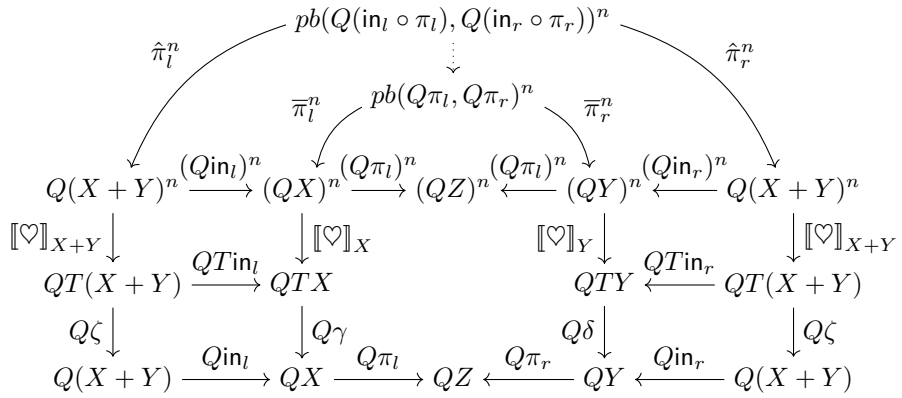
On a single T-coalgebra, the relations $\sim_\Lambda$ and $\sim_{\Lambda_+}$ coincide, but in general they differ.

▶ **Proposition 27.** *For all $x, x' \in X$ and $y \in Y$,*

1. *$\mathbb{X}, x \sim_\Lambda \mathbb{Y}, y$ implies $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$. The implication is strict.*
2. *$\mathbb{X}, x \sim_\Lambda \mathbb{X}, x'$ iff $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{X}, x'$.*

Proof.

Item 1. Let $Z \subseteq X \times Y$ be a Λ-bisimulation between $\mathbb{X}$ and $\mathbb{Y}$. We show that the relation $X + Y \xleftarrow{\mathsf{in}_l \circ \pi_l} Z \xrightarrow{\mathsf{in}_r \circ \pi_r} X + Y$ is a Λ-bisimulation on $\mathbb{X} + \mathbb{Y} = (X + Y, \zeta)$. The proof follows from the commutativity of the diagram below in which $\heartsuit \in \Lambda$ is arbitrary. The commutativity follows from observing that $pb(Q(\mathsf{in}_l \circ \pi_l), Q(\mathsf{in}_r \circ \pi_r))$ with $\hat{\pi}_l \circ Q\mathsf{in}_l$ and $\hat{\pi}_r \circ Q\mathsf{in}_r$ is a competitor to the pullback $(pb(Q\pi_l, Q\pi_r), \overline{\pi}_l, \overline{\pi}_r)$. This yields a mediating map (dashed arrow) such that the upper part of the diagram commutes. The lower, outer parts commute due to naturality of $[\![\heartsuit]\!]$ and the inclusions being T-coalgebra morphisms.



Item 2. ($\Rightarrow$) follows from item 1. For ($\Leftarrow$), assume that Z is a Λ-bisimulation on $\mathbb{X}+\mathbb{X} = (X+X, \zeta)$. We show that $Z' = \{(w, w') \in X \times X \mid \exists i, j \in \{r, l\} : (\mathsf{in}_i(w), \mathsf{in}_j(w')) \in Z\}$ is a Λ-bisimulation on $\mathbb{X}$. First, note that is (U, V) if Z'-coherent, then $(U+U, V+V)$ is Z-coherent. Let $(x, x') \in Z'$, then $(\mathsf{in}_i(x), \mathsf{in}_j(x')) \in Z$, for some $i, j \in \{l, r\}$. Since Z is a Λ-bisimulation, it follows that:

$$\zeta(\mathsf{in}_i(x)) \in [\![\heartsuit]\!]_{X+X}(U+U) \iff \zeta(\mathsf{in}_j(x')) \in [\![\heartsuit]\!]_{X+X}(V+V) \tag{2}$$

To complete the proof, it remains to show that for very $U \subseteq X$

$$\gamma(x) \in [\![\heartsuit]\!]_X[U] \iff \zeta(\mathsf{in}_i(x)) \in [\![\heartsuit]\!]_{X+X}(U+U) \ \ (i = l, r) \tag{3}$$

But this follows from naturality and the fact that inclusion maps are T-coalgebra morphism. Item 2 then follows from (2) and (3). ◀

Due to Proposition 27(1), we define Hennessy-Milner classes of T-coalgebras with respect to $\sim_{\Lambda_+}$.

▶ **Definition 28.** A class $\mathbf{C}$ of T-coalgebras is a *Hennessy-Milner class*, if for every $\mathbb{X}$ and $\mathbb{Y}$ in $\mathbf{C}$, we have $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$ iff $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$.

As a first step towards our main result, we show that the class of finite T-coalgebras is a Hennessy-Milner class. We will use the following terminology. Given a T-coalgebra $\mathbb{X} = (X, \gamma)$, a subset $U \subseteq X$ is *modally coherent* if U is $\equiv_\Lambda$-closed. (Recall that $\equiv_\Lambda$ denotes the modal equivalence relation.) The next lemma provides us with a characterisation of modally coherent sets.

▶ **Lemma 29.** *Let $\mathbb{X}$ be a finite T-coalgebra. For all $U \subseteq X$, U is modally coherent iff U is definable by a modal $\mathcal{L}_\Lambda$-formula.*

Proof. It can be proved using the same line of argumentation as in the proof of [8, Lemma 4.5]. If $U = [\![\varphi]\!]_{\mathbb{X}}$ for some $\varphi \in \mathcal{L}_\Lambda$, then clearly U is modally coherent. For the converse implication, assume U is modally coherent, i.e., U is a union of modal equivalence classes: $U = \bigcup_{i \in I}[x_i]_{\equiv_\Lambda}$. Since X is finite, we may assume that I is finite. For $i, j \in I$ and $i \neq j$, there is a modal $\mathcal{L}_\Lambda$-formula $\delta_{i,j}$ such that $x_i \models \delta_{i,j}$ and $x_j \models \neg\delta_{i,j}$, so by taking $D_i = \{\delta_{i,j} \mid i, j \in I, i \neq j\}$, we have $[x_i]_{\equiv_\Lambda} = \bigcap_{i \in I}[\![D_i]\!] \subseteq X$. Since I is finite, D_i is finite. Defining $\delta_i = \bigwedge D_i$ for each $i \in I$, we then have $U = \bigcup_{i \in I}[\![\delta_i]\!]_{\mathbb{X}}$. Therefore, U is definable by the formula $\delta = \bigvee \delta_i$. ◀

Now, we have the finite version of Hennessy-Milner theorem for Λ-bisimulation.

▶ **Theorem 30.** *Let $\mathbb{X} = (X, \gamma)$ and $\mathbb{Y} = (Y, \delta)$ be finite T-coalgebras, and let Λ be a set of predicate liftings for T.*

1. *For all states $x, x' \in X$: $\mathbb{X}, x \equiv_\Lambda \mathbb{X}, x'$ iff $\mathbb{X}, x \sim_\Lambda \mathbb{X}, x'$.*
2. *For all states $x \in X$ and $y \in Y$: $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$ iff $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$.*

Proof.

Item 1: The direction from right to left has been shown in Proposition 18. For the other direction, we show that $\equiv_\Lambda$ is a Λ-bisimulation. Let $x, x' \in X$ be such that $\mathbb{X}, x \equiv_\Lambda \mathbb{X}, x'$, and let $\heartsuit \in \Lambda$. For simplicity, we just give the argument for unary $[\![\heartsuit]\!]$. The n-ary generalisation is straightforward. Let $U \subseteq X$ be modally coherent. By Lemma 29, U is definable by a $\mathcal{L}_\Lambda$-formula ψ. We therefore have $x \in [\![\heartsuit\psi]\!]_{\mathbb{X}}$ iff $x' \in [\![\heartsuit\psi]\!]_{\mathbb{X}}$ because x and x' are modally equivalent. It follows that $\gamma(x) \in [\![\heartsuit]\!]_X(U)$ iff $\gamma(x') \in [\![\heartsuit]\!]_X(U)$. Hence, $\equiv_\Lambda$ is a Λ-bisimulation on $\mathbb{X}$.

Item 2: Follows from item 1 and the fact that the inclusion maps preserve truth of modal formulas: $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$ iff $\mathbb{X} + \mathbb{Y}, \mathsf{in}_l(x) \sim_\Lambda \mathbb{X} + \mathbb{Y}, \mathsf{in}_r(y)$ iff $\mathbb{X} + \mathbb{Y}, \mathsf{in}_l(x) \equiv_\Lambda$ iff $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$. ◀

We leverage the result for finite T-coalgebras to coalgebras for finitary functors.

▶ **Theorem 31** (Finitary Hennessy-Milner theorem)**.** *Suppose T is a finitary functor, and $\mathbb{X} = (X, \gamma)$, $\mathbb{Y} = (Y, \delta)$ are T-coalgebras.*

1. *For all states $x, x' \in X$: $\mathbb{X}, x \equiv_\Lambda \mathbb{X}, x'$ iff $\mathbb{X}, x \sim_\Lambda \mathbb{X}, x.'$*
2. *For every $x \in X$ and $y \in Y$: $\mathbb{X}, x \equiv_\Lambda \mathbb{Y}, y$ iff $\mathbb{X}, x \sim_{\Lambda_+} \mathbb{Y}, y$.*

Proof.

Item 1: Let $x, x' \in X$ be such that $\mathbb{X}, x \equiv_\Lambda \mathbb{X}, x'$. By [1, Theorem 4.1] there exists a finite subcoalgebra $\mathbb{X}_0 = (X_0, \gamma_0)$ of $\mathbb{X}$ with $x, x' \in X_0$. Since, the inclusion $\mathsf{in}_{X_0} : X_0 \to X$ is a T-coalgebra morphism and hence preserves truth of formulas, it follows that $\mathbb{X}_0, x \equiv_\Lambda \mathbb{X}_0, x'$. By Theorem 30(1) we obtain $\mathbb{X}_0, x \sim_\Lambda \mathbb{X}_0, x'$, and from Proposition 23, using again that in_{X_0} is a T-coalgebra morphism that $\mathbb{X}, x \sim_\Lambda \mathbb{X}, x'$.

Item 2: can be proved using item 1 in a similar way as item 2 of Theorem 30. ◀

5 Discussion and Future Work

We have shown that our notion of Λ-bisimulation gives rise to a Hennessy-Milner theorem, and thus it fits exactly the expressiveness of the modal language. The coherence condition in the definition of Λ-bisimulation is, however, a non-local property as one would need to compute all coherent pairs over the state space in order to verify that two states are Λ-bisimilar. For concrete instances of Λ-bisimulations, it would be desirable to have a local back-and-forth style characterisation, similar to, e.g., the usual ones for Kripke frames, and the zig-zag conditions for Δ-bisimulations over Kripke frames in [6]. Such a local condition would obtain if Λ-bisimilarity could be charaterised in terms of relation liftings. In the case that Λ is separating, respectively monotone, Λ-bisimilarity coincides with precocongruences, respectively GS-Λ-bisimilarity, both of which have a relation lifting characterisation, cf. [8, 7]. We would like to investigate whether approaches such as those of [15, 18] can be used to obtain a relation lifting characterisation of Λ-bisimilarity under weaker conditions.

In [2], a Van Benthem characterisation theorem was proved for contingency logic over neighbourhood frames, that is, over neighbourhood frames, contingency logic is the fragment of first order logic which is invariant under Λ-bisimilarity, where $\Lambda = \{\Delta\}$. We would like to generalise this result and show a coalgebraic version for Λ-bisimilarity, using as

correspondence language *coalgebraic predicate logic (CPL)*, which was introduced in [16] as a first order correspondence language of coalgebraic modal logic.

We hardly explored the notion of Λ-morphisms in the present paper. It would be interesting to know which constructions are possible in the category of T-coalgebras and Λ-morphisms. For example, in [2] it was shown that for $T = \mathcal{N}$ and $\Lambda = \{\Delta\}$, one can construct Λ-quotients, i.e., quotients of T-coalgebras with respect to Λ-bisimilarity. We would like to know whether this is possible, in general. That would mean that we can minimise T-coalgebras with respect to Λ-bisimilarity. Finally, we would also like to know if a final object can be constructed from satisfied theories using techniques along the lines of [12, 20], and whether the Hennessy-Milner theorem for Λ-bisimilarity fits into the more abstract picture where a coalgebraic modal logic is obtained via a dual adjunctions, as in e.g. [11, 10].

References

1. J. Adámek and H. Porst. From varieties of algebras to covarieties of coalgebras. In *CMCS 2000*, volume 44 of *Electr. Notes in Theor. Comp. Sci.*, pages 27–46. Elsevier, 2001.
2. Z. Bakhtiari, H. van Ditmarsch, and H.H. Hansen. Neighbourhood contingency bisimulation. In *ICLA 2017*, volume 10119 of *LNCS*, pages 48–63. Springer, 2017.
3. M. Bílková and M. Dostál. Expressivity of many-valued modal logics, coalgebraically. In *WoLLIC 2016*, volume 9803 of *LNCS*, pages 109–124. Springer, 2016.
4. S. Enqvist. Homomorphisms of coalgebras from predicate liftings. In *CALCO 2013*, volume 8089 of *LNCS*, pages 126–140. Springer, 2013.
5. J. Fan and H. van Ditmarsch. Neighborhood contingency logic. In *ICLA 2015*, volume 8923 of *LNCS*, pages 88–99. Springer, 2015.
6. J. Fan, Y. Wang, and H. van Ditmarsch. Almost necessary. In *AiML 2014*, pages 178–196. College Publications, 2014.
7. D. Gorín and L. Schröder. Simulations and bisimulations for coalgebraic modal logics. In *CALCO 2013*, volume 8089 of *LNCS*, pages 253–266. Springer, 2013.
8. H.H. Hansen, C. Kupke, and E. Pacuit. Neighbourhood structures: Bisimilarity and basic model theory. *Logical Methods in Computer Science*, 5(2:2), 2009.
9. M. Hennessy and R. Milner. Algebraic laws for non-determinism and concurrency. *Journal of the ACM*, 32:137–161, 1985.
10. B. Jacobs and A. Sokolova. Exemplaric expressivity of modal logics. *Journal of logic and computation*, 20(5):1041–1068, 2010.
11. B. Klin. Coalgebraic modal logic beyond sets. *Electronic Notes in Theoretical Computer Science*, 173:177–201, 2007.
12. C. Kupke and R.A. Leal. Characterising behavioural equivalence: Three sides of one coin. In *CALCO 2009*, volume 5728 of *LNCS*, pages 97–112. Springer, 2009.
13. C. Kupke and D. Pattinson. Coalgebraic semantics of modal logics: an overview. *Theoretical Computer Science*, 412(38):5070–5094, 2011.
14. A. Kurz. *Logics for coalgebras and applications to computer science*. PhD thesis, Ludwig-Maximilians-Universität München, 2000.
15. P.B. Levy. Similarity quotients as final coalgebras. In *FoSSaCS 2011*, volume 6604 of *LNCS*, pages 27–41. Springer, 2011.
16. T. Litak, D. Pattinson, K. Sano, and L. Schröder. Coalgebraic predicate logic. In *ICALP 2012*, volume 7392 of *LNCS*, pages 299–311. Springer, 2012.
17. S. Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 2013.
18. J. Marti and Y. Venema. Lax extensions of coalgebra functors and their logic. *Journal of Computer and System Sciences*, 81(5):880–900, 2015.

19 H. Montgomery and R. Routley. Contingency and non-contingency bases for normal modal logics. *Logique et Analyse*, 9:318–328, 1966.

20 L. Moss and I. Viglizzo. Harsanyi type spaces and final coalgebras constructed from satisfied theories. In *CMCS 2004*, volume 106 of *Electr. Notes in Theor. Comp. Sci.*, pages 279–295. Elsevier, 2004.

21 D. Pattinson. Coalgebraic modal logic: Soundness, completeness and decidability of local consequence. *Theoretical Computer Science*, 309(1-3):177–193, 2003.

22 D. Pattinson et al. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame Journal of Formal Logic*, 45(1):19–33, 2004.

23 J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

24 L. Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. *Theoretical Computer Science*, 390:230–247, 2008.

25 J. van Benthem, N. Bezhanishvili, S. Enqvist, and J. Yu. Instantial neighbourhood logic. *Review of Symbolic Logic*, 10(1):116–144, 2017.

Monoidal Company for Accessible Functors*†

Henning Basold[1], Damien Pous[2], and Jurriaan Rot[3]

1 **Radboud University, Nijmegen, The Netherlands**
henning@basold.eu
2 **Univ Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, France**
Damien.Pous@ens-lyon.fr
3 **Radboud University, Nijmegen, The Netherlands**
jrot@cs.ru.nl

Abstract

Distributive laws between functors are a fundamental tool in the theory of coalgebras. In the context of coinduction in complete lattices, they correspond to the so-called compatible functions, which enable enhancements of the coinductive proof technique. Amongst these, the greatest compatible function, called the companion, has recently been shown to satisfy many good properties.

Categorically, the companion of a functor corresponds to the final object in a category of distributive laws. We show that every accessible functor on a locally presentable category has a companion. Central to this and other constructions in the paper is the presentation of distributive laws as coalgebras for a certain functor. This functor itself has again, what we call, a second-order companion. We show how this companion interacts with the various monoidal structures on functor categories. In particular, both the first- and second-order companion give rise to monads. We use these results to obtain an abstract GSOS-like extension result for specifications involving the second-order companion.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases coalgebras, distributive laws, accessible functors, monoidal categories

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.5

1 Introduction

Coalgebras are an abstract tool for defining and studying the semantics of state-based systems [7]. Distributive laws of various kinds play a crucial role in the theory of coalgebras and coinduction. For instance, they are used in (structural) operational semantics [23, 3], for automata constructions [20], and for coinductive proof techniques [6].

In the context of complete lattices, distributive laws for a given functor correspond to functions *compatible* with a given function [15]. Those were introduced to obtain a modular theory of enhancements of the coinductive proof method (up-to techniques): most of the useful enhancements can be presented as compatible functions, and their class is closed under union and composition. In particular, the union of all compatible functions is always a compatible function, the greatest one. This greatest compatible function, called the *companion*, subsumes all compatible functions and is a closure operator [16].

* A full version of the paper is available at https://hal.archives-ouvertes.fr/hal-01529340/, [4].
† This work was funded by the European Research Council (ERC) under the Horizon 2020 programme (CoVeCe, No. 678157) and the Seventh Framework Programme FP7/2007-2013 (QCLS, No. 320571), by the project ANR 12IS02001 PACE and the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX-0007).

The last two authors recently gave a categorical account of the companion of a functor [17]: it can be defined as the final object in the category of distributive laws over that functor; if it exists it is a monad (and the underlying distributive law is that of a monad); and under some conditions, it can be constructed explicitly as the codensity monad of the final sequence of the starting functor. The latter existence result corresponds to a Kleene-like fixpoint theorem, and yields a characterisation of the companion in complete lattices similar to that from Parrow and Weber's work [14]. It is also shown in [17] that the companion of a polynomial functor can be characterised using an abstract notion of causal algebra.

Here we pursue this line of work in another direction, by investigating structural properties and existence of the companion via a Knaster-Tarski-like theorem [10, 22]: accessible functors on a locally presentable category have a final coalgebra [13, 1].

In Section 3 we establish an adjunction between coalgebras for an endofunctor B and distributive laws over B, so that a final coalgebra for B can be obtained from the companion of B. We also recover that the companion is a distributive law of a monad by observing that the (strict) monoidal structure of composition in the category of endofunctors lifts to the category of distributive laws. Monoids for this lifted monoidal structure are precisely distributive laws of monads, and so is the companion (which is a monoid by finality).

Being defined as the largest compatible function, the companion in complete lattices can be obtained as the greatest fixpoint of a carefully chosen functional [16]. We extend this idea categorically in Section 4, by associating to a given endofunctor B a second-order endofunctor $\mathbb{B}$ whose category of coalgebras is isomorphic to the category of distributive laws over B. Slightly more generally, we have the following bijective correspondence:

$$\frac{FB \Rightarrow BG}{F \Rightarrow \mathbb{B}(G)}$$

The second-order functor $\mathbb{B}$ is defined using right Kan extensions. It was used by Street to establish another correspondence, between distributive laws of monads and monad maps [21]. We show that $\mathbb{B}$ is lax monoidal, so that the aforementioned isomorphism actually is an isomorphism of monoidal categories.

To get the existence of the companion, it suffices to show that $\mathbb{B}$ exists and has a final coalgebra; this is where we use accessibility (Section 5). There is a technical subtlety here. Indeed, for size reasons, we restrict $\mathbb{B}$ to the sub-category of κ-accessible functors, for some large enough regular cardinal κ. Doing so, the companion we obtain as a final $\mathbb{B}$-coalgebra depends on κ: it is κ-accessible, and it subsumes only those distributive laws that are κ-accessible. By considering larger and larger cardinals, one can thus obtain a sequence of companions relative to those cardinals. (In small categories, this sequence actually converges.)

Building on those results, we give a new account for some results in structural operational semantics, where one usually considers more permissive notions of distributive laws. For instance, one often works with natural transformations of the shape $\rho\colon FB \Rightarrow BF^\star$, where $F^\star$ is the free monad over F. This is fine because every such natural transformation can be turned into a distributive law $\rho^\sharp\colon F^\star B \Rightarrow BF^\star$.

According to the previous bijection, natural transformations such as the above ρ are in one-to-one correspondence with coalgebras for the composite functor $\mathbb{B}(-^\star)$, which can be considered as coalgebras for $\mathbb{B}$ up to $-^\star$. This observation makes it possible to reuse the theory of up-to techniques to propose a new format (Section 6). Indeed, the second-order functor $\mathbb{B}$ being accessible, it admits a second order companion, $\mathbb{T}$, and one can work with *distributive laws up to* $\mathbb{T}$, that is, natural transformations of type $FB \Rightarrow B\mathbb{T}(F)$.

By lifting another monoidal structure, we prove that $\mathbb{T}$ produces monads: given any functor F, $\mathbb{T}(F)$ is always a monad (but not the free one). We show that starting from a

distributive law up to $\mathbb{T}$ as above, one obtains a distributive law of the monad $\mathbb{T}(F)$ over B. We illustrate the use of these distributive laws up to $\mathbb{T}$ in the stream calculus [19].

When starting with a natural transformation like the above $\rho\colon FB \Rightarrow BF^\star$, for which tools not requiring the second order companion already exist, we show that the two approaches are consistent: they eventually lead to the same F-algebra on the final B-coalgebra. We conjecture that a similar result holds with respect to abstract GSOS specifications [23].

Most omitted proofs can be found in the full version of this abstract [4].

2 Preliminaries

Before we dive into the content of the paper, we introduce some notation and concepts that we use throughout. We use capital, calligraphic letters like $\mathcal{C}, \mathcal{D}, \dots$ to stand for general categories. We write $[\mathcal{C}, \mathcal{D}]$ for the category of functors from $\mathcal{C}$ to $\mathcal{D}$ with natural transformation $\alpha, \beta, \dots$ as morphisms. Given a functor $F\colon \mathcal{C} \to \mathcal{D}$, we denote by $F^*\colon [\mathcal{D}, \mathcal{E}] \to [\mathcal{C}, \mathcal{E}]$ the functor that pre-composes with F, and by $F_*\colon [\mathcal{E}, \mathcal{C}] \to [\mathcal{E}, \mathcal{D}]$ the functor that post-composes with F. If X is an object in $\mathcal{D}$, then we write $K_X\colon \mathcal{C} \to \mathcal{D}$ for the constant functor that maps every object in $\mathcal{C}$ to X. For the sake of clarity, we denote general functors between functor categories by blackboard letters $\mathbb{F}, \mathbb{G}, \mathbb{H}, \dots$ and refer to them as *second-order functors*. Consequently, we call the category of functors between functor categories the *second-order functor category*. Given an endofunctor $B\colon \mathcal{C} \to \mathcal{C}$, we denote the category of B-coalgebras by $\mathsf{coalg}(B)$.

2.1 Locally Presentable Categories and Accessible functors

The construction of the companion in Section 5 crucially uses locally presentable categories and accessible functors thereon. We recall these notions and some of their properties, see [2] for an extensive treatment. Let us first describe locally κ-presentable categories for a regular cardinal κ. A diagram $D\colon \mathcal{I} \to \mathcal{C}$ is said to be *κ-filtered* if the category $\mathcal{I}$ is κ-filtered, that is, if every diagram in $\mathcal{I}$ smaller than κ has a cocone. Whenever the colimit of D exists, it is called κ-filtered as well. Next, we say $X \in \mathcal{C}$ is a *κ-presented* object if the hom-functor $\mathcal{C}(X, -)\colon \mathcal{C} \to \mathsf{Set}$ preserves κ-filtered colimits. Finally, a category $\mathcal{C}$ is *locally κ-presentable* if it is locally small, cocomplete, and there is a set S of κ-presented objects in $\mathcal{C}$ that generates $\mathcal{C}$: every object in $\mathcal{C}$ is a κ-filtered colimit of objects from S.

Central to working with locally presentable categories is the notion of accessible functors. A functor $F\colon \mathcal{C} \to \mathcal{C}$ is *κ-accessible* if it preserves κ-filtered colimits. We denote the category of κ-accessible endofunctors on $\mathcal{C}$ by $[\mathcal{C}, \mathcal{C}]^\kappa$. Note that κ-accessible functors can be composed, thus for a κ-accessible functor $F\colon \mathcal{C} \to \mathcal{C}$, the pre- and post-composition functors F^* and F_* restrict to endofunctors on $[\mathcal{C}, \mathcal{C}]^\kappa$.

We shall mention some important results that we need later. For a locally κ-presentable category $\mathcal{C}$, we denote by $\mathcal{C}_\kappa$ the full subcategory of κ-presentable objects and the inclusion functor by $I\colon \mathcal{C}_\kappa \to \mathcal{C}$. By [13, Proposition 2.1.5], $\mathcal{C}_\kappa$ is *essentially small*, that is, $\mathcal{C}_\kappa$ is equivalent to a small category. Lastly, by [13, Proposition 2.4.3] the pre-composition functor I^* has a left adjoint as in $[\mathcal{C}, \mathcal{C}]^\kappa \underset{\longleftarrow}{\overset{I^*}{\longrightarrow}} [\mathcal{C}_\kappa, \mathcal{C}]$, which also is an adjoint equivalence, see [13, Corollary 2.1.9]. This allows us to represent κ-accessible functors by functors on generators.

2.2 Monoidal Categories and Monads

Monoidal categories are categories that come with a notion of tensor product and a unit for that tensor product. For the purpose of this exposition, we are only interested in *strict monoidal categories*. These are triples $(\mathcal{C}, \otimes, I)$, where $\mathcal{C}$ is a category, $\otimes\colon \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ is a

functor, the *tensor*, and $I \in \mathcal{C}$ is an object, the *unit*. This data is subject to the following equations: $X \otimes (Y \otimes Z) = (X \otimes Y) \otimes Z$, $X \otimes I = X$, $I \otimes X = X$. Since we will only encounter strict monoidal categories, we drop the adjective "strict". A category $\mathcal{D}$ is said to be a *monoidal subcategory* of $\mathcal{C}$ if $\mathcal{D}$ is a subcategory of $\mathcal{C}$, and $\mathcal{D}$ is closed under the tensor product and contains the unit I. Given a functor $F\colon \mathcal{C} \to \mathcal{D}$ between monoidal categories $(\mathcal{C}, \otimes_{\mathcal{C}}, I_{\mathcal{C}})$ and $(\mathcal{D}, \otimes_{\mathcal{D}}, I_{\mathcal{D}})$, we say that F is a *lax monoidal functor* if there is a morphism $\beta\colon I_{\mathcal{D}} \to F(I_{\mathcal{C}})$ and a natural transformation $\alpha\colon \otimes_{\mathcal{D}} \circ (F \times F) \Rightarrow F \circ \otimes_{\mathcal{C}}$. These morphisms must fulfil the following three equations for all objects $X, Y, Z \in \mathcal{C}$: $\alpha_{I_{\mathcal{C}},X} \circ (\beta \otimes_{\mathcal{D}} \mathsf{id}_{FX}) = \mathsf{id}_{FX}$, $\alpha_{X,I_{\mathcal{C}}} \circ (\mathsf{id}_{FX} \otimes_{\mathcal{D}} \beta) = \mathsf{id}_{FX}$ and $\alpha_{X \otimes_{\mathcal{C}} Y,Z} \circ (\alpha_{X,Y} \otimes_{\mathcal{D}} \mathsf{id}_{FZ}) = \alpha_{X,Y \otimes_{\mathcal{C}} Z} \circ (\mathsf{id}_{FX} \otimes_{\mathcal{D}} \alpha_{Y,Z})$. If α and β are both identities, we say that F is a *strict monoidal functor*. Finally, we consider monoidal categories to be isomorphic if there is a strict monoidal isomorphism between them.

The two relevant examples of monoidal structures are given by the different ways functors can be composed. For any category $\mathcal{C}$, there is an obvious monoidal structure on the functor category $[\mathcal{C}, \mathcal{C}]$, defined in terms of functor composition. We denote this structure by $([\mathcal{C}, \mathcal{C}], *, \mathsf{Id})$, where the tensor product on functors $F, G \in [\mathcal{C}, \mathcal{C}]$ is defined by $F * G = F \circ G$, and for natural transformations α and β the tensor $\alpha * \beta$ is given by horizontal composition. If $(\mathcal{D}, \otimes, I)$ is itself a monoidal category, then there is a second way of turning $[\mathcal{D}, \mathcal{D}]$ into a monoidal category, by point-wise tensoring of functors. That is, one defines a tensor product $\otimes'$ by $(F \otimes' G)(D) = F(D) \otimes G(D)$ and on natural transformations by $(\alpha \otimes' \beta)_D = \alpha_D \otimes \beta_D$. The identity is given by the constant functor K_I with $K_I(D) = I$. In this paper we will use the particular instance with $(\mathcal{D}, \otimes, I) = ([\mathcal{C}, \mathcal{C}], *, \mathsf{Id})$ on the second-order functor category; we denote this instance by $([[\mathcal{C}, \mathcal{C}], [\mathcal{C}, \mathcal{C}]], \circledast, K_{\mathsf{Id}})$.

2.3 Monads and Distributive Laws

We will make good use of the folklore phrase "a monad is just a monoid in the category of endofunctors". A monoid in a monoidal category $(\mathcal{C}, \otimes, I)$ is a triple (X, m, e), where $X \in \mathcal{C}$, $m\colon X \otimes X \to X$ and $e\colon I \to X$, such that $m \circ (\mathsf{id} \otimes e) = \mathsf{id}$, $m \circ (e \otimes \mathsf{id}) = \mathsf{id}$ and $m \circ (m \otimes \mathsf{id}) = m \circ (\mathsf{id} \otimes m)$. Accordingly, a monad is a triple (F, μ, η),[1] where F is an endofunctor on $\mathcal{C}$, and $\mu\colon F * F \Rightarrow F$ and $\eta\colon \mathsf{Id} \Rightarrow F$ are natural transformations; the monoid laws are then just the usual monad laws. A *monad map* from (F, μ^F, η^G) to (G, μ^G, η^G) is a natural transformation $F \Rightarrow G$ that makes the evident coherence diagrams commute. Finally note that lax monoidal functors preserve monoids.

The central objects of study in this paper are distributive laws. Given endofunctors $B, F\colon \mathcal{C} \to \mathcal{C}$ on a category $\mathcal{C}$, a *distributive law* is a natural transformation $FB \Rightarrow BF$. Such a distributive law induces a lifting of F to $\mathsf{coalg}(B)$. If (F, μ^F, η^F) is a monad, we say that $\rho\colon FB \Rightarrow BF$ is a *distributive law of a monad (over B)*, provided that $\rho \circ \eta^F B = B\eta^F$ and $\rho \circ \mu^F B = B\mu^F \circ \rho F \circ F\rho$ hold. On rare occasions, we need to generalise distributive laws to *asymmetric distributive laws*, which are natural transformations $FB \Rightarrow BG$ for another endofunctor G on $\mathcal{C}$. Similarly, if G carries a monad structure (G, μ^G, η^G), we say that $\rho\colon FB \Rightarrow BG$ is an *asymmetric distributive law of monads (over B)*, whenever $\rho \circ \eta^F B = B\eta^G$ and $\rho \circ \mu^F B = B\mu^G \circ \rho G \circ F\rho$ hold.[2]

[1] We divert from the usual order (F, η, μ) for denoting monads, since for monoids and monoidal categories the multiplication comes first.

[2] Note that Street [21] refers in this situation to (B, ρ) as a monad functor.

3 The Companion of a Functor

We define the notion of companion, and show several of its properties. Throughout this section, let $\mathcal{F}$ be a full, monoidal subcategory of $([\mathcal{C},\mathcal{C}], *, \mathsf{Id})$, and $B\colon \mathcal{C} \to \mathcal{C}$ a functor in $\mathcal{F}$. The reader can safely assume $\mathcal{F} = [\mathcal{C},\mathcal{C}]$. The slight generalisation to subcategories is required for the material in Section 5, where we restrict to categories of accessible functors.

▶ **Definition 1.** The category $\mathsf{DL}(B)$ of distributive laws (w.r.t. the subcategory $\mathcal{F}$) is defined as follows. An object is a pair (F,λ) where $F\colon \mathcal{C} \to \mathcal{C}$ is a functor in $\mathcal{F}$ and $\lambda\colon FB \Rightarrow BF$ is a natural transformation. A *morphism of distributive laws* from (F,λ) to (G,ρ) is a natural transformation $\delta\colon F \Rightarrow G$ such that $\rho \circ \delta B = B\delta \circ \lambda$, see [18, 24, 11, 9]. The *companion* of B is the final object of $\mathsf{DL}(B)$, if it exists. We typically denote the companion of B by (T^B, τ^B), or (T,τ) if B is clear from the context. Given an object (F,λ) in $\mathsf{DL}(B)$, we write $\lambda^\dagger\colon F \Rightarrow T$ for the unique morphism obtained by finality.

3.1 Final Coalgebra from the Companion

Suppose the underlying category $\mathcal{C}$ has an initial object $\mathsf{0}$. Consider the functor $\mathsf{ev}_\mathsf{0}\colon \mathsf{DL}(B) \to \mathsf{coalg}(B)$, defined on objects as $\mathsf{ev}_\mathsf{0}(F,\lambda) = \lambda_\mathsf{0} \circ F!_{B\mathsf{0}}\colon F\mathsf{0} \to BF\mathsf{0}$. In [17], we showed that applying ev_0 to the companion yields a final B-coalgebra. Here we generalise the situation to subcategories of $[\mathcal{C},\mathcal{C}]$ and show a stronger result: the functor ev_0 has a left adjoint.

A coalgebra $f\colon X \to BX$ is a distributive law of the constant functor K_X over B. If K_X is an object of $\mathcal{F}$, for each X, then this gives rise to a functor, which is left adjoint to ev_0.

▶ **Theorem 2.** *Suppose that $\mathcal{F}$ contains all constant functors K_X for X in $\mathcal{C}$. Then K extends to a functor $K\colon \mathsf{coalg}(B) \to \mathsf{DL}(B)$ which is a left adjoint of ev_0.*

$$\mathsf{coalg}(B) \underset{\mathsf{ev}_\mathsf{0}}{\overset{K}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathsf{DL}(B)$$

Hence, if (T,τ) is the companion, then $\mathsf{ev}_\mathsf{0}(T,\tau)$ is a final B-coalgebra.

It follows from the above theorem that not every functor B has a companion.

3.2 Monoidal Structure of Distributive Laws

In [17], we showed that the companion, if it exists, is always a monad. It turns out that this result can be phrased slightly more generally based on the monoidal structure of $\mathcal{F}$ given by composition. The main observation is that the monoidal structure of $[\mathcal{C},\mathcal{C}]$ lifts to $\mathsf{DL}(B)$, and that a monoid in $\mathsf{DL}(B)$ corresponds to a monad with a distributive law over B.

▶ **Theorem 3.** *The category $\mathsf{DL}(B)$ is strict monoidal, with tensor product $*$ given by*

$$(F,\lambda) * (G,\rho) = (FG, \lambda G \circ F\rho)$$

and the unit by trivial distributive law $(\mathsf{Id}, \mathsf{id}\colon B \Rightarrow B)$. An object (F,λ) is a monoid in $(\mathsf{DL}(B), *, \mathsf{id})$ if and only if F is a monad and λ a distributive law of that monad over B.

Proof. For an object (F,λ) of $\mathsf{DL}(B)$, (F,μ,η) is a monoid iff
1. η is a morphism from $(\mathsf{Id},\mathsf{id})$ to (F,λ), i.e., $B\eta = \lambda \circ \eta B$;
2. μ is a morphism from $(F,\lambda) * (F,\lambda)$ to (F,λ), i.e., $\lambda \circ \mu B = B\mu \circ \lambda F \circ F\lambda$;
3. (F,μ,η) is a monoid in $\mathcal{F}$.

The first two items are the axioms of distributive laws of monad over functor, the third is equivalent to (F, μ, η) being a monad. ◀

It is straightforward that the final object of any monoidal category, if it exists, is a monoid. Instantiating this to $\mathsf{DL}(B)$ and applying Theorem 3, we obtain the following result. The first item appeared in [17], for $\mathcal{F} = [\mathcal{C}, \mathcal{C}]$.

▶ **Corollary 4.** *Suppose (T, τ) is the companion of B.*

1. *There are unique $\eta\colon \mathsf{Id} \Rightarrow T$ and $\mu\colon TT \Rightarrow T$ such that (T, μ, η) is a monad and $\tau\colon TB \Rightarrow BT$ is a distributive law of this monad over B.*
2. *For any (F, λ) in $\mathsf{DL}(B)$, if F is a monad and λ a distributive law of that monad over B, then $\lambda^\dagger\colon F \Rightarrow T$ is a monad map.*

4 Distributive Laws as Coalgebras

In this section we work again with an endofunctor B on a monoidal subcategory $\mathcal{F}$ of $[\mathcal{C}, \mathcal{C}]$. An important idea underlying the current paper is that distributive laws over B can be characterised as coalgebras for a certain functor $\mathbb{B}\colon \mathcal{F} \to \mathcal{F}$. To obtain $\mathbb{B}$, suppose that B has a (global) right Kan extension $\mathsf{Ran}_B(-)\colon \mathcal{F} \to \mathcal{F}$, that is, a right adjoint to the pre-composition functor B^*. This gives us a bijective correspondence

$$\frac{FB \Rightarrow G}{F \Rightarrow \mathsf{Ran}_B G}\ \textbf{(Kan)}$$

natural in F and G. We obtain the correspondence between coalgebras and distributive laws from **(Kan)** by taking $G = BF$. More precisely, we compose Ran_B and B_* to the functor

$$\mathbb{B} = \mathsf{Ran}_B(B-) : \mathcal{F} \to \mathcal{F}\,. \tag{1}$$

We call this functor the *familiar of B*. From **(Kan)** we get the announced correspondence.

▶ **Lemma 5.** *The category $\mathsf{coalg}(\mathbb{B})$ is isomorphic to $\mathsf{DL}(B)$.*

Proof. The isomorphism is immediate by the natural bijection **(Kan)**. Note that the functor $\mathsf{coalg}(\mathbb{B}) \to \mathsf{DL}(B)$ is given by transposing along the adjunction, that is, it maps $\lambda\colon F \Rightarrow \mathbb{B}(F)$ to $\epsilon_F \circ \lambda B\colon FB \Rightarrow BF$, where ϵ_F is the counit of the Kan extension $\mathbb{B}(F)$. ◀

In particular, the companion of B is the final $\mathbb{B}$-coalgebra.

▶ **Example 6.** Let $b\colon L \to L$ be a monotone function on a complete lattice L. The associated $\mathbb{B}\colon [L, L] \to [L, L]$ was given in [16] by $\mathbb{B}(f) = \bigvee_{gb \le bf} g$. The correspondence in Lemma 5 means that f is b-compatible (i.e., $fb \le bf$) iff it is a post-fixed point of $\mathbb{B}$. The standard pointwise computation of right Kan extensions by limits gives another characterisation:

$$\mathbb{B}(f)(x) = \bigwedge_{x \le b(y)} bf(y)\,.$$

Street [21] considers the functor $\mathbb{B}$ in the context of monads, to obtain the following correspondence between monad maps and distributive laws. We use this result in Section 6.

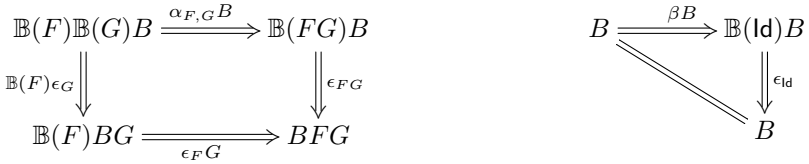
▶ **Lemma 7.** *If G is a monad, then $\mathbb{B}(G)$ is a monad as well. Moreover, given monads F, G there is a one-to-one correspondence:*

$$\frac{FB \Rightarrow BG \quad \textit{asymm. d.l. of monads over } B}{F \Rightarrow \mathbb{B}(G) \quad \textit{monad map}}\ \textbf{(Street)}$$

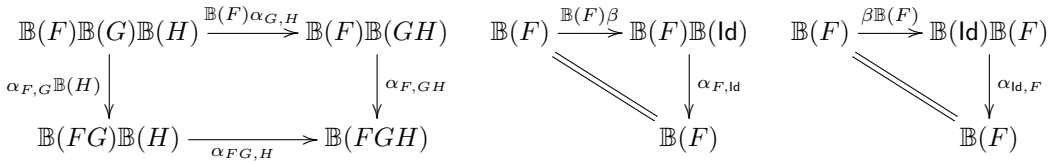
We complete the picture by showing that the familiar $\mathbb{B}$ is lax monoidal.

▶ **Theorem 8.** *The functor $\mathbb{B}$ is a lax monoidal endofunctor on $(\mathcal{F}, *, \mathsf{Id})$.*

Proof. To show that $\mathbb{B}$ is lax monoidal, we use that for each $H\colon \mathcal{C} \to \mathcal{C}$, $(\mathbb{B}(H), \epsilon_H)$ is a right Kan extension of BH along B. This allows us to define for functors $F, G\colon \mathcal{C} \to \mathcal{C}$ the mediators $\alpha_{F,G}\colon \mathbb{B}(F)\mathbb{B}(G) \Rightarrow \mathbb{B}(FG)$ and $\beta\colon \mathsf{Id} \Rightarrow \mathbb{B}(\mathsf{Id})$ as the unique natural transformations such that the following two diagrams commute.



Naturality of α (in F and G) follows via the adjunction of the right Kan extension from naturality of $\epsilon_F G \circ \mathbb{B}(F)\epsilon_G$ (in F and G), we skip the details. It remains to prove the coherence axioms for α and β. Given $F, G, H\colon \mathcal{C} \to \mathcal{C}$, we thus need to prove:



All these diagrams commute by appealing to the universal property of the Kan extensions. ◀

The above theorem allows us to turn $\mathsf{coalg}(\mathbb{B})$ into a monoidal category, with tensor product defined by $(F, \lambda) * (G, \rho) = (FG, \alpha_{F,G} \circ (\lambda * \rho))$ and with unit $\mathsf{Id} = (\mathsf{Id}, \beta)$. This monoidal structure is the same, modulo an isomorphism, as the monoidal structure of $\mathsf{DL}(B)$ given by composition, as we show in the following lemma.

▶ **Lemma 9.** *The monoidal category $(\mathsf{coalg}(\mathbb{B}), *, \mathsf{Id})$ is isomorphic to $(\mathsf{DL}(B), *, \mathsf{Id})$. Hence, $((F, \lambda), \mu, \eta)$ is a monoid in $\mathsf{DL}(B)$ if and only if $((F, \lambda^*), \mu, \eta)$ is a monoid in $\mathsf{coalg}(B)$, where λ^* is the coalgebra associated to λ by the isomorphism.*

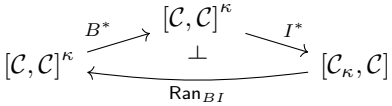
Proof. It suffices to prove that the functor from $\mathsf{coalg}(\mathbb{B})$ to $\mathsf{DL}(B)$ in Lemma 5 is strict monoidal. Hence, we have to prove for $\lambda\colon F \Rightarrow \mathbb{B}(F)$ and $\rho\colon G \Rightarrow \mathbb{B}(G)$ that transposing $(FG, \alpha_{F,G} \circ (\lambda * \rho))$ is the same as the monoidal product of the transpose of (F, λ) and the transpose of (G, ρ) in $\mathsf{DL}(B)$. In turn, this follows by a routine calculation. ◀

5 Constructing the Companion of an Accessible Functor

We show now that the companion of an accessible functor generally exists. More concretely, we assume that $B\colon \mathcal{C} \to \mathcal{C}$ is a κ-accessible functor on a locally κ-presentable category $\mathcal{C}$. We instantiate the subcategory $\mathcal{F}$ of Section 3 to the category $[\mathcal{C}, \mathcal{C}]^\kappa$ of κ-accessible functors. For the sake of clarity, let us denote the associated category of distributive laws by $\mathsf{DL}_\kappa(B)$, and refer to the companion, the final object in $\mathsf{DL}_\kappa(B)$, as the *κ-companion*. Further, we denote the associated familiar of B by $\mathbb{B}_\kappa$, and call it the *κ-familiar*. The presentation of distributive laws over B as coalgebras for $\mathbb{B}_\kappa$ allows us to construct in Theorem 12 the κ-companion as a final $\mathbb{B}_\kappa$-coalgebra. We begin by showing that the κ-familiar $\mathbb{B}_\kappa$ exists.

▶ **Lemma 10.** *The functor $B^*\colon [\mathcal{C},\mathcal{C}]^\kappa \to [\mathcal{C},\mathcal{C}]^\kappa$ has a right adjoint, given by $\mathsf{Ran}_{BI}(-I)$.*

Proof. Recall the inclusion functor $I\colon \mathcal{C}_\kappa \to \mathcal{C}$ and consider the right Kan extension Ran_{BI}:



This Kan extension exists and is computed pointwise by the standard limit formula, see, e.g., [12], using that $\mathcal{C}_\kappa$ is essentially small and $\mathcal{C}$ is complete [2, Corollary 1.28]. The desired adjunction is given by the bijective correspondence below, which is natural both in F and G.

$$\dfrac{\dfrac{FB \Rightarrow G}{FBI \Rightarrow GI}}{F \Rightarrow \mathsf{Ran}_{BI}GI}\ \textbf{(Kan)}$$

The upper correspondence (natural in FB and G) comes from the fact that I^* is part of an equivalence, and the lower one (natural in F and GI) from the above right Kan extension. ◀

Using Lemma 10, we can show that the familiar $\mathbb{B}_\kappa$ exists. Recall that we defined in (1) the familiar as the composition of the right adjoint $\mathsf{Ran}_{BI}(-I)$ and B_*, thus we have $\mathbb{B}_\kappa(F) = \mathsf{Ran}_{BI}(BFI)$. Lemma 5 asserts that $\mathsf{coalg}(\mathbb{B}_\kappa) \cong \mathsf{DL}_\kappa(B)$. The problem of finding a κ-companion now reduces to finding a final $\mathbb{B}_\kappa$-coalgebra, for which we can use standard tools: any accessible functor on a locally presentable category has a final coalgebra.

▶ **Lemma 11.** *The functor $\mathbb{B}_\kappa$ is accessible.*

Proof. The functor B_* is accessible since B is, and colimits in functor categories are computed pointwise. The functor $\mathsf{Ran}_{BI}(-)$ is accessible, since it is a right adjoint on locally presentable categories, which in turn follows from the adjoint functor theorem for locally presentable categories [2, Theorem 1.66]. Since the composition of accessible functors is again accessible, we obtain that $\mathbb{B}_\kappa$ is accessible. ◀

Note that the above lemma states that $\mathbb{B}_\kappa$ is accessible, and *not* that it is κ-accessible. The adjoint functor theorem guarantees accessibility of $\mathsf{Ran}_{BI}(-)$ only for a cardinal that is potentially larger than κ.

▶ **Theorem 12.** *The functor B has a κ-companion.*

Proof. The category $[\mathcal{C},\mathcal{C}]^\kappa$ is locally presentable, since it is equivalent to the category $[\mathcal{C}_\kappa,\mathcal{C}]$, which is locally presentable [2]. Since $\mathbb{B}_\kappa$ is accessible by Lemma 11, $\mathsf{coalg}(\mathbb{B}_\kappa)$ has a final object, see [13] or [1, Theorem 4.2.12]. This final object gives the κ-companion through the isomorphism between $\mathsf{coalg}(\mathbb{B}_\kappa)$ and $\mathsf{DL}_\kappa(B)$. ◀

▶ **Example 13.**

1. Any complete lattice L is locally presentable, and any monotone function $b\colon L \to L$ is accessible (both for a sufficiently large cardinal). Hence, we obtain the companion for any such b, and thereby recover the corresponding result in [16].
2. The finite powerset functor $\mathcal{P}_\omega$ on Set is ω-accessible, hence it has an ω-companion. Of course, ω can be replaced here by any regular cardinal.
3. More generally, define the class of Kripke-polynomial functors (on Set) as the least class that contains $\mathcal{P}_\omega$, the constant functors K_A and exponent functors $(-)^A$ for every set A, and which is closed under finite products, non-empty coproducts and composition. It is well-known that every Kripke polynomial functor is accessible, see, e.g., [7, Lemma 4.6.8], hence any Kripke-polynomial functor has a κ-companion, for some κ.

▶ Remark. Any constant functor K_X, for X an object of $\mathcal{C}$, is ω-accessible. By Theorem 2, the κ-companion of B yields a final B-coalgebra by evaluation on an initial object of $\mathcal{C}$.

6 Second-Order Companion and Distributive Laws Up-To

The construction of the previous section can be iterated to obtain "higher-order" companions: By Lemma 11, $\mathbb{B}_\kappa$ is again an accessible functor on the locally presentable category $[\mathcal{C},\mathcal{C}]^\kappa$. Hence, by Theorem 12, the functor $\mathbb{B}_\kappa$ itself has a companion $\mathbb{T}_\kappa$. More generally, we assume in this section that both the familiar $\mathbb{B}$ of B and the companion $\mathbb{T}$ of $\mathbb{B}$ exist. We refer in the sequel to $\mathbb{T}$ as the *second-order companion*.

Such a second-order companion turned out to be a useful tool for proving soundness of up-to techniques in the context of complete lattices [16]. Here, we use second-order companions to propose general GSOS-type specifications presented by distributive laws in Section 6.2. We first provide some background on such specifications.

In the theory of coalgebras, distributive laws $\rho\colon FB \Rightarrow BF$ are frequently used as an abstract specification format, where F represents the syntax (typically F is a polynomial functor representing an algebraic signature), B the type of behaviour and ρ the semantics (see [8] for an overview). For specific instances of B (and F), such natural transformations correspond to concrete syntactic rule formats. It is customary in this approach to start from more permissive types of distributive laws, that also correspond to more general rule formats, such as $\rho\colon FB \Rightarrow B(F+\mathsf{Id})$, or $\rho\colon FB \Rightarrow BF^\star$, where $F^\star$ is the free monad over F. An even more general type is given by the celebrated *abstract GSOS specifications*, of the form $\rho\colon F(B\times\mathsf{Id}) \Rightarrow BF^\star$, which are briefly discussed at the end of the paper.

Such natural transformations can typically be extended to distributive laws, possibly with additional structure. In particular, a natural transformation $\rho\colon FB \Rightarrow BF^\star$ corresponds uniquely to a distributive law $\rho^\sharp\colon F^\star B \Rightarrow BF^\star$ of the free monad $F^\star$ over B. This is typically proved by appealing to initiality of algebras, see, e.g., [3]. We can give an elegant proof by using the familiar as follows, where the second step uses that $\mathbb{B}(F^\star)$ is a monad (Lemma 7).

$$\dfrac{\dfrac{\dfrac{FB \Rightarrow BF^\star}{F \Rightarrow \mathbb{B}(F^\star)}\,\textbf{(Kan)}}{F^\star \Rightarrow \mathbb{B}(F^\star) \quad \text{monad map}}\,\textbf{(Free)}}{F^\star B \Rightarrow BF^\star \quad \text{d.l. of monad over functor}}\,\textbf{(Street)}$$

In the remainder of this section we consider a more general type of natural transformation: that of the form $\rho\colon FB \Rightarrow B\mathbb{T}(F)$. We refer to it as a *distributive law up to* $\mathbb{T}$. The main result is that $\mathbb{T}(F)$ carries a monad structure, and that every such ρ extends to a distributive law of this monad over B. The approach is at a high level similar to the one for the case $FB \Rightarrow BF^\star$ (c.f. Theorem 15), but it is significantly more involved, as $\mathbb{T}(F)$ is not a free monad F in general (note, for instance, that it depends on B). Later in this section, we show that distributive laws up to $\mathbb{T}$ properly generalise distributive laws of the form $FB \Rightarrow BF^\star$.

Throughout this section, we let $\mathcal{F}$ be a full monoidal subcategory of $([\mathcal{C},\mathcal{C}],*,\mathsf{Id})$ and $\mathcal{S}$ be a full subcategory of $[\mathcal{F},\mathcal{F}]$ such that

1. $\mathcal{S}$ is a monoidal subcategory of $([\mathcal{F},\mathcal{F}],*,\mathsf{Id})$, that is, $\mathcal{S}$ is closed under composition and contains the identity functor;
2. $\mathcal{S}$ is a monoidal subcategory of $([\mathcal{F},\mathcal{F}],\circledast,K_{\mathsf{Id}})$, that is, $\mathcal{S}$ contains the constant functor K_{Id}, and if $\mathbb{F},\mathbb{G}\in\mathcal{S}$, then the functor $\mathbb{F}\circledast\mathbb{G}$ given by $F\mapsto\mathbb{F}(F)\mathbb{G}(F)$ is in $\mathcal{S}$;
3. the familiar $\mathbb{B}$ exists and is an element of $\mathcal{S}$;
4. the familiar $\mathbb{B}$ has a companion $\mathbb{T}$.

The category $\mathsf{DL}(B)$ is given by distributive laws over B (with functors in $\mathcal{F}$), and the category $\mathsf{DL}(\mathbb{B})$ is given by distributive laws over $\mathbb{B}$ (with functors in $\mathcal{S}$). The main instance of interest is given by the accessible functors, for which $\mathbb{B}$ and $\mathbb{T}$ exist whenever B is accessible:

▶ **Lemma 14.** *Let $B\colon \mathcal{C} \to \mathcal{C}$ be κ-accessible. Let $\mathcal{F} = [\mathcal{C},\mathcal{C}]^\kappa$ and $\mathcal{S} = [\mathcal{F},\mathcal{F}]^\lambda$, where $\lambda \geq \kappa$ is a regular cardinal such that $\mathbb{B}_\kappa$ is λ-accessible. These $\mathcal{F}$ and $\mathcal{S}$ meet assumptions **1.-4.***

6.1 Second-Order Companion and Monads

An important feature of the companion is that it is a monad, which allows us to collapse multiple uses of the companion. This result lifts to the second-order companion $\mathbb{T}$ in two ways. First, by Theorem 3 the category $\mathsf{DL}(\mathbb{B})$ of distributive laws over $\mathbb{B}$ inherits a monoidal structure from $(\mathcal{S}, *, \mathsf{Id})$. This gives rise to a monad structure $(\mathbb{T}, \mu, \eta)$ on $\mathbb{T}$, see Corollary 4. We denote the associated distributive law of that monad over $\mathbb{B}$ by $\pi\colon \mathbb{T}\mathbb{B} \Rightarrow \mathbb{B}\mathbb{T}$. Second, more interestingly, $\mathbb{T}$ has a monoid structure in $(\mathcal{S}, \circledast, K_{\mathsf{Id}})$. This is proved in Theorem 15 by using that $(\mathcal{S}, \circledast, K_{\mathsf{Id}})$ lifts to $\mathsf{DL}(\mathbb{B})$. The monoid structure neatly encapsulates the fact that $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ is a monad, for every functor $F\colon \mathcal{C} \to \mathcal{C}$ in $\mathcal{F}$ (Corollary 16).

▶ **Theorem 15.** *The monoidal structure of $(\mathcal{S}, \circledast, K_{\mathsf{Id}})$ lifts to $\mathsf{DL}(\mathbb{B})$. This yields a monoid $(\mathbb{T}, \dot{\mu}\colon \mathbb{T} \circledast \mathbb{T} \to \mathbb{T}, \dot{\eta}\colon K_{\mathsf{Id}} \to \mathbb{T})$ on the second-order companion.*

Proof. We use that $\mathbb{B}$ is lax monoidal (Theorem 8) with mediators $\beta\colon \mathsf{Id} \Rightarrow \mathbb{B}(\mathsf{Id})$ and $\alpha_{F,G}\colon \mathbb{B}(F)\mathbb{B}(G) \Rightarrow \mathbb{B}(FG)$ (natural in $F, G \in \mathcal{F}$). Now, given $(\mathbb{F}, \lambda)$ and $(\mathbb{G}, \rho)$ in $\mathcal{S}$, for the tensor $(\mathbb{F}, \lambda) \circledast (\mathbb{G}, \rho)$ we have to provide a distributive law of type $(\mathbb{F} \circledast \mathbb{G})\mathbb{B} \Rightarrow \mathbb{B}(\mathbb{F} \circledast \mathbb{G})$, which we define on a component F as:

$$(\mathbb{F} \circledast \mathbb{G})\mathbb{B}(F) = \mathbb{F}\mathbb{B}(F)\mathbb{G}\mathbb{B}(F) \xRightarrow{\lambda_F * \rho_F} \mathbb{B}\mathbb{F}(F)\mathbb{B}\mathbb{G}(F) \xRightarrow{\alpha_{\mathbb{F}(F),\mathbb{G}(F)}} \mathbb{B}(\mathbb{F}(F)\mathbb{G}(F)) \ .$$

The distributive law for the unit K_{Id} is defined by:

$$K_{\mathsf{Id}}\mathbb{B}(F) = \mathsf{Id} \xRightarrow{\beta} \mathbb{B}(\mathsf{Id}) = \mathbb{B}K_{\mathsf{Id}}(F)$$

Naturality and the axioms of monoidal categories are routine calculations. ◀

Given a functor F in $\mathcal{F}$, we get a strict monoidal functor $\mathsf{ev}_F\colon \mathcal{S} \to \mathcal{F}$ by letting $\mathsf{ev}_F(\mathbb{G}) = \mathbb{G}(F)$. Since monoidal functors preserve monoids, $\mathbb{T}(F)$ is a monoid in $\mathcal{F}$, i.e., a monad.

▶ **Corollary 16.** *For every functor F in $\mathcal{F}$, $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ is a monad.*

We now prove that any distributive law $\lambda\colon FB \Rightarrow BF$ gives rise to a distributive law of the monad $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ over B (Corollary 18 below). To do so, we first extend ev_F to a strict monoidal functor $\mathsf{ev}_{(F,\lambda)}\colon \mathsf{DL}(\mathbb{B}) \to \mathsf{DL}(B)$ (Theorem 17). Let $\rho\colon \mathbb{G}\mathbb{B} \Rightarrow \mathbb{B}\mathbb{G}$ be a distributive law. We obtain a lifting $\overline{\mathbb{G}}\colon \mathsf{coalg}(\mathbb{B}) \to \mathsf{coalg}(\mathbb{B})$ by $\overline{\mathbb{G}}_\rho(G, \gamma) = \rho_G \circ \mathbb{T}(\gamma)$. From this, we construct a distributive law over B by applying the following transformation to λ:

$$\dfrac{\dfrac{\dfrac{FB \Rightarrow BF}{F \Rightarrow \mathbb{B}(F)}\,\textbf{(Kan)}}{\mathbb{G}(F) \Rightarrow \mathbb{B}\mathbb{G}(F)}\,\textbf{(Lifting } \overline{\mathbb{G}}\textbf{)}}{\mathbb{G}(F)B \Rightarrow B\mathbb{G}(F)}\,\textbf{(Kan)}$$

This construction gives rise to a functor $\mathsf{ev}_{(F,\lambda)}\colon \mathsf{DL}(\mathbb{B}) \to \mathsf{DL}(B)$.

▶ **Theorem 17.** *The functor* $\mathsf{ev}_{(F,\lambda)}$ *is strict monoidal, from* $(\mathsf{DL}(\mathbb{B}), \circledast, K_{\mathsf{Id}})$ *to* $(\mathsf{DL}(B), *, \mathsf{Id})$.

Proof. The functor $\mathsf{ev}_{(F,\lambda)}$ decomposes as a functor $\mathsf{DL}(\mathbb{B}) \to \mathsf{coalg}(\mathbb{B})$ followed by the (monoidal) isomorphism $\mathsf{coalg}(\mathbb{B}) \cong \mathsf{DL}(B)$ from Lemma 9. It is easy to check that the functor $\mathsf{DL}(\mathbb{B}) \to \mathsf{coalg}(\mathbb{B})$ is strict monoidal as well. ◀

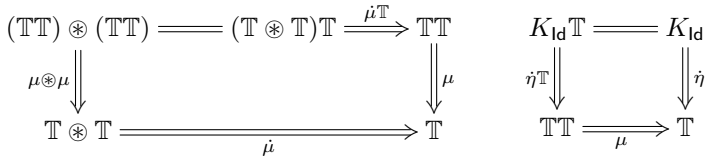
By applying this to the monoid on the second-order companion from Theorem 15, we obtain:

▶ **Corollary 18.** *For an object* (F, λ) *in* $\mathsf{DL}(B)$, $\mathsf{ev}_{(F,\lambda)}(\mathbb{T}, \pi)$ *is a distributive law of the monad* $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ *over* B.

6.2 Distributive Laws up to $\mathbb{T}$

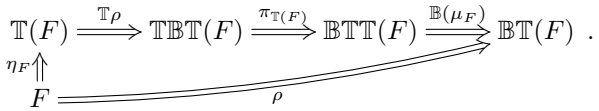
Now we show how to extend distributive laws of the form $FB \Rightarrow B\mathbb{T}(F)$ to distributive laws of the *monad* $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ over B. The idea is to first transpose such a distributive law to $\rho\colon F \Rightarrow \mathbb{B}\mathbb{T}(F)$, and then apply, by using the distributive law $\pi\colon \mathbb{T}\mathbb{B} \Rightarrow \mathbb{B}\mathbb{T}$ of the second-order companion, what is sometimes called the generalised powerset construction [20]. Lemma 20 shows how this extension interacts with the monad $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$. For its proof, we need the following lemma, which relates the two monoid structures on $\mathbb{T}$.

▶ **Lemma 19.** *The following diagrams commute.*



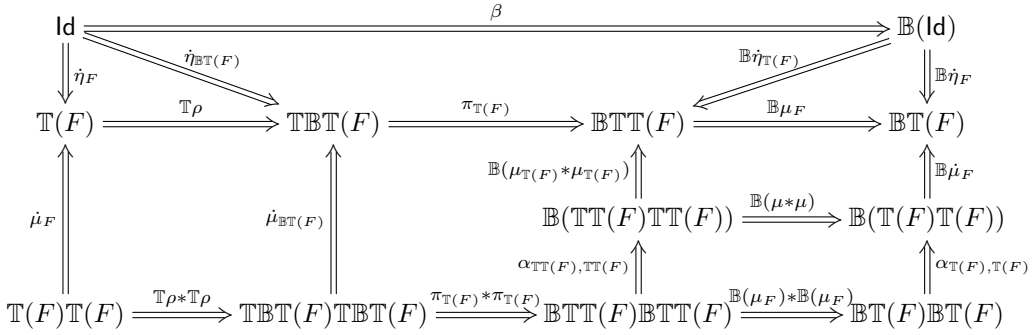
Proof. Use finality of $\mathbb{T}$ in $\mathsf{DL}(\mathbb{B})$ to prove that the axioms hold in $\mathsf{DL}(\mathbb{B})$. This follows once we show distributivity (on the right) of composition over $\circledast$ in $\mathsf{DL}(\mathbb{B})$, i.e., the equalities in the diagrams should hold in $\mathsf{DL}(\mathbb{B})$ as well. This is a straightforward exercise. ◀

▶ **Lemma 20.** *Extend* $\rho\colon F \Rightarrow \mathbb{B}\mathbb{T}(F)$ *to the natural transformation* $\rho^\sharp\colon \mathbb{T}(F) \Rightarrow \mathbb{B}\mathbb{T}(F)$, *which is given by the transformation in the top line in the following diagram.*



Then $((\mathbb{T}(F), \rho^\sharp), \dot{\mu}_F, \dot{\eta}_F)$ *is a monoid in* $\mathsf{coalg}(\mathbb{B})$.

Proof. We only need to prove that $\dot{\eta}_F$ and $\dot{\mu}_F$ are morphisms of the correct type in $\mathsf{coalg}(\mathbb{B})$. This is given by commutativity of the upper and lower parts of the following diagram.



The diagram commutes clockwise, starting from the triangle on the top left, by naturality of $\dot{\eta}$, definition of $\dot{\eta}$, Lemma 19 twice, naturality of α, definition of $\dot{\mu}$ and naturality of $\dot{\mu}$. ◀

The following theorem states the main extension result.

▶ **Theorem 21.** *Let $\rho\colon FB \Rightarrow B\mathbb{T}(F)$ with F in $\mathcal{F}$. There exists a distributive law $\bar{\rho}\colon \mathbb{T}(F)B \Rightarrow B\mathbb{T}(F)$ of the monad $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ over B such that $\bar{\rho} \circ \eta_F B = \rho$.*

Proof. We have the following inference.

$$\dfrac{\dfrac{\dfrac{FB \Rightarrow B\mathbb{T}(F)}{F \Rightarrow \mathbb{B}\mathbb{T}(F)}\,\textbf{(Kan)}}{\mathbb{T}(F) \Rightarrow \mathbb{B}\mathbb{T}(F) \quad \text{monoid in } \mathsf{coalg}(\mathbb{B}) \text{ with structure } (\dot{\mu}_F, \dot{\eta}_F)}\,\textbf{(Lemma 20)}}{\mathbb{T}(F)B \Rightarrow B\mathbb{T}(F) \quad \text{monoid in } \mathsf{DL}(B) \text{ with structure } (\dot{\mu}_F, \dot{\eta}_F)}\,\textbf{(Lemma 9)}$$

The conclusion corresponds to a distributive law of the monad $(\mathbb{T}(F), \dot{\mu}_F, \dot{\eta}_F)$ over B (Theorem 3). The resulting natural transformation $\bar{\rho}$ satisfies $\bar{\rho} \circ \eta_F B = \rho$. ◀

6.3 A Toolkit for Distributive Laws up to $\mathbb{T}$

As explained before, distributive laws of the shape $FB \Rightarrow BF$ give rise to F-algebras on the final B-coalgebra. A problem with such specifications is that they must define all the involved operations. For instance, the definition of the convolution product on streams requires the simultaneous definition of point-wise addition [19]. Theorem 21 above makes it possible to work instead with distributive laws up to $\mathbb{T}$, of the shape $FB \Rightarrow B\mathbb{T}(F)$. This is convenient in practice as it makes it possible to define distributive laws, and thus operations on the final coalgebra, in a modular way. This is demonstrated in the example on streams below.

Given a functor F, which describes the signature of the operations to be defined, a specification can often be given by finding another functor G and a natural transformation $FB \Rightarrow BG$ that directly describe the behaviour of the operations. To turn such a natural transformation into a distributive law up to $\mathbb{T}$, an explicit or extensional knowledge of $\mathbb{T}$ is usually not required; it suffices to find a natural transformation $G \Rightarrow \mathbb{T}(F)$. Below we provide a small toolkit to construct such natural transformations in a modular way. This toolkit should be seen as a first stepping stone for a typed calculus of specification up to $\mathbb{T}$, which allows the modular construction of coalgebraic specifications.

Recall that T denotes the (first-order) companion of B; we have the following natural transformations:

- $t\colon T \Rightarrow \mathbb{T}(F)$ — If $\mathcal{F}$ has an initial object $\mathsf{0}$ and contains the constant functors K_X for all X. In that case $T = \mathbb{T}(\mathsf{0})$ (Theorem 2), and $t = \mathbb{T}(!_F)\colon \mathbb{T}(\mathsf{0}) \Rightarrow \mathbb{T}(F)$.
- $b\colon B \Rightarrow \mathbb{T}(F)$ — Obtained as the composition of the unique distributive law morphism $B \Rightarrow T$ (from $\mathsf{id}\colon BB \Rightarrow BB$ to the companion T) and t.
- $\eta_F\colon F \Rightarrow \mathbb{T}(F)$ — Unit of the monad on $\mathbb{T}$.
- $\dot{\eta}_F\colon \mathsf{Id} \Rightarrow \mathbb{T}(F)$ — Unit of the monad on $\mathbb{T}(F)$.
- $\dot{\mu}_F\colon \mathbb{T}(F)\mathbb{T}(F) \Rightarrow \mathbb{T}(F)$ — Multiplication of the monad on $\mathbb{T}(F)$.

The monad multiplication $\dot{\mu}_F$ allows us to combine $G \Rightarrow \mathbb{T}(F)$ and $H \Rightarrow \mathbb{T}(F)$ by composition: $GH \Rightarrow \mathbb{T}(F)\mathbb{T}(F) \Rightarrow \mathbb{T}(F)$. Moreover, the monad on $\mathbb{T}(F)$ allows us to extend any $G \Rightarrow \mathbb{T}(F)$ to a monad map $G^\star \Rightarrow \mathbb{T}(F)$, if the free monad G^* of G exists. In particular, we obtain:

- $s\colon F^\star \Rightarrow \mathbb{T}(F)$ — the unique monad map such that $s \circ \iota = \eta_F$, where $\iota\colon F \Rightarrow F^\star$ is the canonical map from F to the associated free monad $F^\star$ and η is the unit of the monad $\mathbb{T}$.

The natural transformation t allows us to reuse distributive laws up to $\mathbb{T}$ in a modular fashion: Any distributive law $\lambda\colon GB \Rightarrow BG$ (or even $\lambda\colon GB \Rightarrow B\mathbb{T}(G)$) gives rise to a natural transformation $\lambda^\dagger\colon G \Rightarrow T$, which can be composed with t to get $t \circ \lambda^\dagger\colon G \Rightarrow \mathbb{T}(F)$.

We can exploit the above toolkit to show that the following types of natural transformations are all (encodable as) instances of distributive laws up to $\mathbb{T}$.

- $FB \Rightarrow BF$ — By using $\eta_F\colon F \Rightarrow \mathbb{T}(F)$.
- $FB \Rightarrow B(F + \mathsf{Id})$ — By using $[\eta_F, \dot{\eta}_F]\colon F + \mathsf{Id} \Rightarrow \mathbb{T}(F)$.
- $FB \Rightarrow BF^\star$ — By using $s\colon F^\star \Rightarrow \mathbb{T}(F)$.
- $FB \Rightarrow B(F + B)^\star$ — By using the unique monad map $(F + B)^\star \Rightarrow \mathbb{T}(F)$ that extends $[\eta_F, b]\colon F + B \Rightarrow \mathbb{T}(F)$.
- $FB \Rightarrow B(F+B+T)^\star$ — Similar to the previous case but with $[\dot{\eta}_F, b, t]\colon F+B+T \Rightarrow \mathbb{T}(F)$.

Each of the above natural transformations gives rise to a $\rho\colon FB \Rightarrow B\mathbb{T}(F)$ and hence, by Theorem 21, extends to a distributive law $\overline{\rho}\colon \mathbb{T}(F)B \Rightarrow B\mathbb{T}(F)$.

In the beginning of the section, we recalled that any $\rho\colon FB \Rightarrow BF^\star$ extends to a distributive law $\rho^\sharp\colon F^\star B \Rightarrow BF^\star$. We show in the following theorem that this is generalised by the extension of distributive laws up to $\mathbb{T}$ from Theorem 21. To this end, we establish a morphism of distributive laws. Intuitively, this means that the results of the two extensions behave the same, as far as $F^\star$ is concerned.

▶ **Theorem 22.** *Extend $\rho\colon FB \Rightarrow BF^\star$ to $\rho^\sharp\colon F^\star B \Rightarrow BF^\star$ and $\overline{(Bs \circ \rho)}\colon \mathbb{T}(F)B \Rightarrow B\mathbb{T}(F)$. Then s is a morphism of distributive laws: $Bs \circ \rho^\sharp = \overline{(Bs \circ \rho)} \circ sB$.*

As a consequence, the F-algebra obtained on the final B-coalgebra from $\rho^\sharp$ coincides with that obtained from $\overline{(Bs \circ \rho)}$.[3]

Example on streams

Let $B\colon \mathsf{Set} \to \mathsf{Set}$ be $BX = \mathbb{R} \times X$, where $\mathbb{R}$ is the set of real numbers. The (carrier of the) final B-coalgebra is given by the set of streams $\mathbb{R}^\omega$. The head and tail of a stream σ are denoted by $\sigma(0)$ and σ' respectively. The following equations define binary operations $+$, $\times$ a unary operation $(-)^*$ and constants $[r]$ for each $r \in \mathbb{R}$ on streams [19]:

$$\begin{aligned} [r](0) &= r & [r]' &= [0] \\ (\sigma + \tau)(0) &= \sigma(0) + \tau(0) & (\sigma + \tau)' &= \sigma' + \tau' \\ (\sigma \times \tau)(0) &= \sigma(0) \times \tau(0) & (\sigma \times \tau)' &= \sigma' \times \tau + [\sigma(0)] \times \tau' \\ (\sigma^*)(0) &= 1 & (\sigma^*)' &= \sigma' \times \sigma^* \end{aligned}$$

We model these using distributive laws up to $\mathbb{T}$, starting with $(-)^*$. Let $SX = X$ and $MX = X \times X$ be the functors that represent the arity of $(-)^*$ and $\times$. Then we define:

$$\begin{aligned} &\rho^*\colon SB \Rightarrow BM(\mathsf{Id} + SB) \\ &\rho^*_X(r, x) = (1, (x, (r, x))) \end{aligned}$$

Alternatively, one can use a natural transformation of the form $SB \Rightarrow B(M + S + B)^*$, but the above type reflects more directly the concrete definition. Notice that the functor B on the right hand side signals the occurrence of σ on the right-hand side of the concrete

[3] Assuming a final coalgebra (Z, z), every distributive law $\gamma\colon GB \Rightarrow BG$ gives rise to a coalgebra $(GZ, \gamma_Z \circ Gz)$ and thus to a G-algebra by finality of (Z, z); when G is $F^\star$, resp. $\mathbb{T}(F)$, this G-algebra can be turned into an F-algebra by precomposing with ι_Z, resp. $\dot{\eta}_F$.

definition of $(\sigma^*)'$. Using the toolkit, which we presented before, it is easy to give a natural transformation $\kappa\colon M(\mathsf{Id} + SB) \Rightarrow M\mathbb{T}(S)$, so we obtain

$$B\kappa \circ \rho^*\colon SB \Rightarrow BM\mathbb{T}(S)\,.$$

This natural transformation extends to a distributive law up to $\mathbb{T}$ once we provide semantics of the product, in the form of a natural transformation $M \Rightarrow \mathbb{T}(S)$.

Again, we can give a precise type for the semantics of the product.

$$\rho^\times\colon MB \Rightarrow BP(M(\mathsf{Id} + B) + M(K_{\mathbb{R}} + \mathsf{Id}))$$
$$\rho^\times_X((r,x),(s,y)) = (r \times s, ((x,(s,y)),(r,y)))$$

Here, $P(X) = X \times X$ models the arity of the sum operator and $K_{\mathbb{R}}$ that of the $[r]$'s for $r \in \mathbb{R}$. Also this semantics could also be presented as a natural transformation of the form $\rho^\times\colon MB \Rightarrow B(P + M + K_{\mathbb{R}})^\star$. Either way, we still need natural transformations $K_{\mathbb{R}} \Rightarrow \mathbb{T}(M)$ and $P \Rightarrow \mathbb{T}(M)$ to complete the specification.

For the sum and constants, we define $\rho^+\colon PB \Rightarrow BP$ and $\rho^{[-]}\colon K_{\mathbb{R}} \Rightarrow BK_{\mathbb{R}}$ by

$$\rho^+_X((r,x),(s,y)) = (r+s, x+y) \qquad \rho^{[-]}(r) = (r,0)$$

Since these are plain distributive laws, we directly obtain natural transformations $P \Rightarrow T$ and $K_{\mathbb{R}} \Rightarrow T$ that, by composing with t from the toolkit, extend to natural transformations $P \Rightarrow \mathbb{T}(M)$ and $K_{\mathbb{R}} \Rightarrow \mathbb{T}(M)$. The toolkit allows us now to combine them into a natural transformation $\delta\colon P(M(\mathsf{Id} + B) + M(K_{\mathbb{R}} + \mathsf{Id})) \Rightarrow \mathbb{T}(M)$, from which we obtain

$$B\delta \circ \rho^\times\colon MB \Rightarrow B\mathbb{T}(M)\,.$$

Since this is a distributive law up to $\mathbb{T}$ we obtain a natural transformation $M \Rightarrow T$ from Theorem 21 and finality of the companion T. Composing with t gives a natural transformation of the form $M \Rightarrow \mathbb{T}(S)$, which we use to complete ρ^* to a distributive law up to $\mathbb{T}$ of the form $SB \Rightarrow B\mathbb{T}(S)$. Finally, again by Theorem 21, we get a distributive law of $\mathbb{T}(S)$ over B.

Abstract GSOS

An *abstract GSOS specification* is a natural transformation of the form $\rho\colon F(B \times \mathsf{Id}) \Rightarrow BF^\star$. It is not directly clear how to encode abstract GSOS as a distributive law up to $\mathbb{T}$, because of the product with the identity functor in the domain. This problem is reflected in the previous concrete example by the problem of modelling, for instance, the occurrence of σ on the right-hand side of the equation for $(\sigma^\star)'$ as a distributive law. There, we solved the problem by using that distributive laws up to $\mathbb{T}$ permit the use of B on the right-hand side, because of the natural transformation $b\colon B \Rightarrow T$.

The following is an attempt to use this idea to encode an arbitrary abstract GSOS ρ as a distributive law up to $\mathbb{T}$:

$$FB \xRightarrow{F\langle B\eta, b\rangle} F(B \times \mathsf{Id})T \xRightarrow{\rho T} BF^\star T \xRightarrow{BsT} B\mathbb{T}(F)T \xRightarrow{B\mathbb{T}(F)t} B\mathbb{T}(F)\mathbb{T}(F) \xRightarrow{B\dot{\mu}_F} B\mathbb{T}(F)$$

We conjecture that this distributive law up to $\mathbb{T}$ encodes the behaviour of ρ, in the sense that they define the same F-algebras on the final B-coalgebra. Such a result has been shown for stream systems ($BX = A \times X$ on Set) through an explicit construction of a distributive law in [5], and more abstractly based on the companion for polynomial functors in [17]. The current approach would generalise it to accessible functors. However, we do not currently know if the above construction is indeed correct, and leave this as an open problem.

Acknowledgements. We thank Bart Jacobs for valuable comments on monoidal categories.

References

1. J. Adamek, S. Milius, and L Moss. Initial algebras and terminal coalgebras: a survey. Preliminary version, 2010. URL: https://www.tu-braunschweig.de/Medien-DB/iti/survey_full.pdf.
2. J. Adamek and J. Rosicky. *Locally Presentable and Accessible Categories*. Cambridge Tracts in Mathematics. Cambridge University Press, 1994. URL: https://books.google.nl/books?id=iXh6rOd7of0C.
3. F. Bartels. *On generalised coinduction and probabilistic specification formats*. PhD thesis, CWI, Amsterdam, April 2004.
4. Henning Basold, Damien Pous, and Jurriaan Rot. Monoidal company for accessible functors (full version, with proofs). URL: https://hal.archives-ouvertes.fr/hal-01529340/.
5. Filippo Bonchi, Matias Lee, and Jurriaan Rot. Bisimilarity of open terms in stream GSOS. In *FSEN*, 2017. To appear.
6. Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. Coinduction up-to in a fibrational setting. In *CSL-LICS*, pages 20:1–20:9. ACM, 2014. URL: http://arxiv.org/abs/1401.6675, doi:10.1145/2603088.2603149.
7. Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in T.C.S.* Cambridge University Press, 2016. doi:10.1017/CBO9781316823187.
8. B. Klin. Bialgebras for structural operational semantics: An introduction. *Theoretical Computer Science*, 412(38):5043–5069, 2011. doi:10.1016/j.tcs.2011.03.023.
9. Bartek Klin and Beata Nachyla. Presenting morphisms of distributive laws. In *CALCO*, volume 35 of *LIPIcs*, pages 190–204. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.CALCO.2015.190.
10. B. Knaster. Un théorème sur les fonctions d'ensembles. *Annales de la Société Polonaise de Mathématiques*, 6:133–134, 1928.
11. Marina Lenisa, John Power, and Hiroshi Watanabe. Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. *Electronic Notes in Theoretical Computer Science*, 33:230–260, 2000. doi:10.1016/S1571-0661(05)80350-0.
12. S. MacLane. *Categories for the working mathematician*. Springer, 1998.
13. M. Makkai and R. Paré. *Accessible Categories: The Foundations of Categorical Model Theory*, volume 104 of *Contemporary mathematics - American Mathematical Society*. American Mathematical Society, 1989.
14. Joachim Parrow and Tjark Weber. The largest respectful function. *Logical Methods in Computer Science*, 12(2), 2016. doi:10.2168/LMCS-12(2:11)2016.
15. Damien Pous. Complete lattices and up-to techniques. In *APLAS*, volume 4807 of *LNCS*, pages 351–366. Springer, 2007. doi:10.1007/978-3-540-76637-7_24.
16. Damien Pous. Coinduction all the way up. In *LICS*, pages 307–316. ACM, 2016. doi:10.1145/2933575.2934564.
17. Damien Pous and Jurriaan Rot. Companions, codensity and causality. In *FoSSaCS*, volume 10203 of *LNCS*, pages 106–123, 2017. doi:10.1007/978-3-662-54458-7_7.
18. John Power and Hiroshi Watanabe. Combining a monad and a comonad. *Theoretical Computer Science*, 280(1-2):137–162, 2002. doi:10.1016/S0304-3975(01)00024-X.
19. Jan J. M. M. Rutten. A coinductive calculus of streams. *Mathematical Structures in Computer Science*, 15(1):93–147, 2005. doi:10.1017/S0960129504004517.
20. Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Logical Methods in Computer Science*, 9(1), 2013. doi:10.2168/LMCS-9(1:9)2013.

21 Ross Street. The formal theory of monads. *J. of Pure and Applied Algebra*, 2(2):149–168, 1972. doi:10.1016/0022-4049(72)90019-9.

22 A. Tarski. A Lattice-Theoretical Fixpoint Theorem and its Applications. *Pacific Journal of Mathematics*, 5(2):285–309, June 1955.

23 D. Turi and G. D. Plotkin. Towards a mathematical operational semantics. In *LICS*, pages 280–291. IEEE, 1997. doi:10.1109/LICS.1997.614955.

24 Hiroshi Watanabe. Well-behaved translations between structural operational semantics. *Electronic Notes in Theoretical Computer Science*, 65(1):337–357, 2002. doi:10.1016/S1571-0661(04)80372-4.

On Path-Based Coalgebras and Weak Notions of Bisimulation*†

Harsh Beohar[1] and Sebastian Küpper[2]

1 **Theoretical Computer Science Group, Universität Duisburg-Essen, Germany**
harsh.beohar@uni-due.de
2 **Theoretical Computer Science Group, Universität Duisburg-Essen, Germany**
sebastian.kuepper@uni-due.de

Abstract

It is well known that the theory of coalgebras provides an abstract definition of behavioural equivalence that coincides with strong bisimulation across a wide variety of state-based systems. Unfortunately, the theory in the presence of so-called silent actions is not yet fully developed. In this paper, we give a coalgebraic characterisation of branching (delay) bisimulation in the context of labelled transition systems (fully probabilistic systems). It is shown that recording executions (up to a notion of stuttering), rather than the set of successor states, from a state is sufficient to characterise the respected bisimulation relations in both cases.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases Paths, Executions, Branching bisimulation, Coalgebras

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.6

1 Introduction

Since its inception, coalgebra-based modelling of systems provides a simple and abstract definition of behavioural equivalence that coincides with the so-called strong bisimulation relations across a wide variety of dynamical systems (see [19] for an introduction). Two states are said to be behaviourally equivalent if they are mapped to a common point by a coalgebra homomorphism. Unfortunately, the theory in the presence of so-called *silent* actions is not yet well developed, albeit some general constructions (with varying level of generality) characterising Milner's weak bisimulation [16] are proposed in the literature (see, for instance, [8, 9, 10, 12, 21] and the references therein).

Another refinement of strong bisimulation is *branching bisimulation* proposed by Glabbeck and Weijland [25], which is the coarsest equivalence (in the Glabbeek spectrum [23]) preserving the branching structure of a state [24]. In this context, we are unaware of any prior work that captured branching bisimulation in the framework of coalgebras. Moreover, a natural notion of behavioural equivalence should preserve the branching structure of a state just like strong bisimulation does in the absence of silent action.

Bonchi et al. [7] have considered silent transitions coalgebraically by removing them all together by considering the labels as words rather than single letters. This approach is not useful when characterising branching bisimulation (or even weak bisimulation) because not

* A full version of the paper is available at https://arxiv.org/abs/1705.08715, [6].

† This work has been carried out as part of the "Behavioural Equivalences: Environmental Aspects, Metrics and Generic Algorithms" (BEMEGA) project supported by Deutsche Forschungsgemeinschaft (DFG).

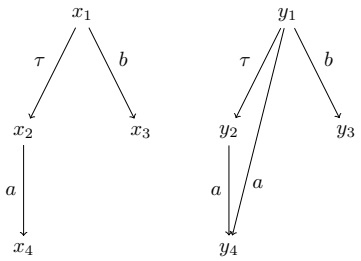


Figure 1 The states x_1, y_1 are weakly bisimilar, but not branching bisimilar.

all silent transitions can always be removed from the system without violating the transfer properties of branching (weak) bisimulation. In [8, 9, 10, 12], weak bisimulation is captured in two phases: first, a given coalgebra is transformed into a coalgebra (possibly over a different base category) which captures the "saturation" effect of a silent action; second, it is shown that the notion of behavioural equivalence on this transformed coalgebra coincides with weak bisimulation on the corresponding dynamical system. In [21] the authors used Aczel-Mendler style formulation of strong bisimulation in the latter step.

Nevertheless, it is well known that the saturation step (i.e. adding strong a-transitions for each weak a-transition to the transition system) is not sound with respect to branching bisimulation even in the case of labelled transition systems [25]; thus, a different approach to characterise it is required. The reason is that τ-steps can enable or disable choice in an observable behaviour, something that is hidden by the saturation step. For weak bisimulation this point is irrelevant; however, branching bisimulation (which is finer than weak bisimulation) requires that τ-steps which are required to answer an observable action may only lead to states that are still in bisimulation relation with the original state. For instance, consider the states x_1, y_1 described in Figure 1. They are *not* branching bisimilar because the transition $y_1 \xrightarrow{a} y_4$ can be simulated by the transitions $x_1 \xrightarrow{\tau} x_2 \xrightarrow{a} x_4$, but the intermediate state x_2 cannot be related with the state y_1 because y_1 can fire a b-transition which x_2 cannot simulate.

Our research hypothesis is that recording executions (up to a notion of stuttering) generated by a state (instead of the set of successor states) is sufficient to capture branching bisimulation across different classes of systems. Stated differently, it is the set of executions (not the set of successor states) which specifies the branching structure of a state in the presence of silent actions. In particular, we will substantiate this claim for the class of labelled transition systems and fully probabilistic systems in this paper.

In lieu of the above hypothesis, we restructure the classical coalgebraic machinery in the following way. We begin by studying a notion of *paths* on an arbitrary set X (denoted $\mathrm{Path}(X)$) in Section 2, which is general enough to specify the executions of a labelled transition system and a fully probabilistic system. Intuitively, a path on X can be viewed as a finite sequence that alternates between the elements of X and an action in the alphabet $A_\tau = A \uplus \{\tau\}$, where $\tau \notin A$ is the silent action. Now for every path $p \in \mathrm{Path}(X)$ there is a unique stutter invariant path $p^\dagger \in \mathrm{Path}(X)$ associated with it, which intuitively can be constructed by removing the τ self-loops. This is reminiscent of a coloured trace from [25], which is obtained from a concrete coloured trace in a process graph whose nodes are labelled by a fixed set of colours. In the sequel, stutter invariance induces an equivalence relation $\sim$ on the set $\mathrm{Path}(X)$ whose quotient is denoted as $\mathrm{Path}_\sim(X)$. Furthermore, it turns out that both the mappings $\mathrm{Path}(X), \mathrm{Path}_\sim(X)$ are endofunctors on the category of sets **Set**.

Now what is missing in our approach is the type of dynamics (also known as the branching type in the theory of coalgebras). For instance, a labelled transition system can be viewed as a coalgebra of type $\mathcal{P} \circ (A_\tau \times \mathrm{id})$ over the base category **Set**. Here $\mathcal{P}$ is the covariant powerset functor and $A_\tau \times \mathrm{id}$ is the product functor whose left component is fixed. In other words, the branching type of labelled transition system is nondeterministic. Therefore, to characterise branching bisimulation, we consider coalgebras of type $\mathcal{P} \circ \mathrm{Path}_\sim$ over the base category **Set**. In Section 3, we show that behavioural equivalence in this coalgebra coincides with the traditional branching bisimulation relation [25]. Moreover, this framework can also be used to characterise the weak bisimulation, delay bisimulation, and eta-bisimulation relations; however, for reasons of space, this is worked out in [6].

Nevertheless, the situation is not so straightforward in the case of a fully probabilistic system. Often such systems are modelled as coalgebras of type $\mathcal{D} \circ (A_\tau \times \mathrm{id})$ over the base category **Set**, where $\mathcal{D}$ is the sub-distribution functor. It turns out that one needs a notion of measurable space and a measure on the set of maximal executions[1] in order to define branching (weak) bisimulation relations over the states of a fully probabilistic system (cf. [4, 22]). Thus, it is natural to consider fully probabilistic systems as 'weighted' coalgebras of type $\mathcal{G} \circ (A_\tau \times \mathrm{id})$ over the base category of measurable spaces **Meas**. Here, $\mathcal{G}$ is the well-known Giry monad of probability measures [17].

In Section 5, just like in the discrete case, we consider coalgebras of type $\mathcal{G} \circ \mathrm{Path}_\sim$ over the base category **Meas** to characterise probabilistic delay bismulation, which was mistakenly [20] called probabilistic branching bisimulation in [21, 22]. The crux of the matter is in defining $\mathrm{Path}(X)$ and $\mathrm{Path}_\sim(X)$ as endofunctors on the category **Meas**. In other words, we need to resolve the following issues: first, which subsets of $\mathrm{Path}(X)$ and $\mathrm{Path}_\sim(X)$ are measurable; second, whether $\mathrm{Path}(X) \xrightarrow{\mathrm{Path}(f)} \mathrm{Path}(Y)$ (for a given $X \xrightarrow{f} Y$ in **Set**) is a measurable function or not; third, constructions of measures on the sets $\mathrm{Path}(X)$ and $\mathrm{Path}_\sim(X)$. These issues are explored in Section 4, for which some preliminary knowledge on topology, domain theory, and measure theory is required. In Section 6, we discuss future directions for research and present some concluding remarks. An extended version of this paper containing all the complete proofs pertaining to each section can be found in [6].

2 Preliminaries

This section is devoted to formally introduce a notion of path and stutter equivalent path on a set X, which will be used throughout the paper. As mentioned earlier, a path on X can be intuitively viewed as a finite sequence that alternates between the elements of the set X and an action in the alphabet A_τ. However, we abstain from this operational view in favour of Definition 1 to reason about paths from a functional perspective.

Let A_τ^* be the set of finite words with $\varepsilon \in A_\tau^*$ denoting the empty sequence. We write $\preceq$ to denote the prefix ordering on words and let $\downarrow\sigma = \{\sigma' \mid \sigma' \preceq \sigma\}$.

▶ **Definition 1.** A *path* p on a set X is a function whose codomain is X and domain is the set of all prefixes of some word in A_τ^*.

Let $\mathrm{Path}(X)$ be the set of all paths on a given set X. Then, this lifts to an endofunctor on the category of sets **Set** by letting: $\mathrm{Path}(f)(p) = f \circ p$, for every function $X \xrightarrow{f} Y$.

[1] An execution of a fully probabilistic system is *maximal* if it is an infinite execution or it stops in a state with the sum of probabilities of all outgoing transitions as 0.

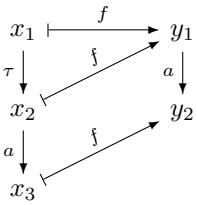


Figure 2 Two systems having same branching structure.

Every path $p \in \mathrm{Path}(X)$ has a *trace* associated with it. Moreover, every path $p \in \mathrm{Path}(X)$ reaches a *last* element from the set X. Symbolically, we write

$$\mathrm{trace}(p) = \max \mathrm{dom}(p) \quad \text{and} \quad \mathrm{last}(p) = p(\mathrm{trace}(p)).$$

▶ **Proposition 2.** *Let* $X \xrightarrow{f} Y$ *be a function. Then, for every path* $p \in \mathrm{Path}(X)$ *we have* $\mathrm{trace}(p) = \mathrm{trace}(fp)$ *and* $f\mathrm{last}(p) = \mathrm{last}(fp)$.

The above proposition states that the functor Path preserves the trace of a path, which is quite strong for our purpose. To exemplify this, consider two labelled transition systems and a function between the states as shown in Figure 2. Since the τ-step from the state x_1 does not disable any choice of observable action offered by x_1, we would like to declare the states x_1 and x_2 as equivalent. In other words, we would like to assert that f is a homomorphism between the coalgebras $(X = \{x_1, x_2, x_3\}, \alpha)$ and $(Y = \{y_1, y_2\}, \beta)$, where the functions α, β return all the generated executions. However, we note that this is not the case because the f-image of the execution $p = \langle x_1\ \tau\ x_2\ a\ x_3\rangle \in \alpha(x_1)$ is $\langle y_1\ \tau\ y_1\ a\ y_2\rangle$ which is not an execution from y_1. Thus, the key observation is to relate the executions of two systems up to stuttering, which leads to the following definition.

▶ **Definition 3.** A function ϕ with $\mathrm{dom}(\phi) = \mathrm{dom}(p)$ and $\mathrm{cod}(\phi) = A_\tau^*$ is a *stutter basis* for a path $p \in \mathrm{Path}(X)$ if it can be constructed inductively by the following rules:

1. $\phi(\varepsilon) = \varepsilon$.
2. if $\sigma'\tau \in \mathrm{dom}(p)$ and $p(\sigma'\tau) = p(\sigma')$ then $\phi(\sigma'\tau) = \phi(\sigma')$.
3. if $\sigma'\tau \in \mathrm{dom}(p)$ and $p(\sigma'\tau) \neq p(\sigma')$ then $\phi(\sigma'\tau) = \phi(\sigma')\tau$.
4. if $\sigma' a \in \mathrm{dom}(p)$ and $a \in A$ then $\phi(\sigma' a) = \phi(\sigma')a$.

As an example, consider a path $p = \langle x_1\ \tau\ x_1\ a\ x_2\rangle$. Then, the function ϕ defined as $\phi(\tau) = \phi(\varepsilon) = \varepsilon$ and $\phi(\tau a) = a$ is a stutter basis ϕ for the path p. However, if $p = \langle x_1\ \tau\ x_2\ a\ x_3\rangle$ with $x_1 \neq x_2$, then $\phi = \mathrm{id}$ is a stutter basis for p.

▶ **Theorem 4.** *For any path there is a unique stutter basis.*

▶ **Lemma 5.** *Let* ϕ *be the stutter basis for a path* $p \in \mathrm{Path}(X)$ *with* $\mathrm{dom}(p) = \downarrow\sigma$, *for some* $\sigma \in A_\tau^*$. *Then,* $\phi(\downarrow\sigma) = \downarrow\phi(\sigma)$.

▶ **Definition 6.** Given a path $p \in \mathrm{Path}(X)$ and its corresponding stutter basis ϕ, then a function $\phi(\mathrm{dom}(p)) \xrightarrow{p^\dagger} X$ is the *stutter invariant* path relative to p if $p^\dagger \circ \phi = p$.

Notice that for the function $p^\dagger$ to be a path its domain should be a prefix closed subset of a word, which follows directly from Lemma 5.

The notion of stutter invariant path induces an equivalence relation on the set of all paths as follows. Two paths $p, q \in \mathrm{Path}(X)$ are said to be *stutter equivalent*, denoted $p \sim q$,

if and only if they have the identical stutter invariant path, i.e., $p^\dagger = q^\dagger$. Let $\mathrm{Path}_\sim(X)$ be the set of all the paths up to stutter equivalence. This lifts to a functor as well:

$$\mathrm{Path}_\sim(f)[p]_\sim = [f \circ p]_\sim \qquad \text{for any } p \in \mathrm{Path}(X) \text{ and } X \xrightarrow{f} Y. \tag{1}$$

To prove that the above map is well-defined, we need the following lemma.

▶ **Lemma 7.** *For any $p \in \mathrm{Path}(X)$ and any $X \xrightarrow{f} Y$, we have $f \circ p^\dagger \sim f \circ p$.*

▶ **Theorem 8.** *The mapping in* (1) *is well defined and* $\mathrm{Path}_\sim$ *is an endofunctor on* **Set**.

Notation We write π_X for the quotient map that maps a path $p \in \mathrm{Path}(X)$ to $[p]_\sim$.
We end this subsection with few properties on (stutter equivalent) paths.

▶ **Lemma 9.** *Let $p \in \mathrm{Path}(X), q \in \mathrm{Path}(Y)$, and $X \xrightarrow{f} Y$.*

1. *If $fp \sim q$ and $\mathrm{trace}(q) \in \tau^* a \tau^*$ then $q(\varepsilon) = fp(\varepsilon) \wedge \mathrm{trace}(p) \in \tau^* a \tau^*$.*
2. $\mathrm{last}(p) = \mathrm{last}(p^\dagger)$.
3. *If $fp \sim q$, then $f(\mathrm{last}(p)) = \mathrm{last}(q)$.*
4. $\pi_Y \circ \mathrm{Path}(f) = \mathrm{Path}_\sim(f) \circ \pi_X$.
5. *Let $V_i \subseteq \mathrm{Path}_\sim(Y)$ (for $i \in I$) be a family of pairwise disjoint sets. Then,*

$$\mathrm{Path}_\sim(f)^{-1}\Big(\bigcup_{i\in I} V_i\Big) \;=\; \bigcup_{i\in I} \mathrm{Path}_\sim(f)^{-1}(V_i) \;\; .$$

Coalgebraic preliminaries

▶ **Definition 10.** Let $\mathbf{C}$ be a category and let $\mathbf{C} \xrightarrow{F} \mathbf{C}$ be an endofunctor. An *F-coalgebra* over the base category $\mathbf{C}$ is a tuple (X, α), where X is an object in $\mathbf{C}$ and $X \xrightarrow{\alpha} FX$ is an arrow in $\mathbf{C}$. Given two objects (X, α) and (Y, β), an F-coalgebra homomorphism is an arrow $X \xrightarrow{f} Y$ in $\mathbf{C}$ such that $Ff \circ \alpha = \beta \circ f$.

▶ **Definition 11.** Let $\mathbf{C}$ be a concrete category over the category of sets **Set**, i.e., there is a faithful functor $\mathbf{C} \xrightarrow{|\_|}$ **Set**. Let (X, α) be an F-coalgebra over the concrete category $\mathbf{C}$. Then, two points $x, x' \in |X|$ are said to be *F-behaviourally equivalent* if and only if there is an F-coalgebra (Y, β) and an F-coalgebra homomorphism $X \xrightarrow{f} Y$ such that $f(x) = f(x')$.

In Section 3, we will let $\mathbf{C} = \mathbf{Set}$ and $|\_| = \mathrm{id}$; however, in Section 5, we will let $\mathbf{C} = \mathbf{Meas}$ and the faithful functor $|\_|$ to be the forgetful functor which forgets the sigma algebras associated with the underlying sets.

3 Branching bisimulation on labelled transition systems

The goal is to characterise branching bisimulation of Glabbeek and Weijland [25] using a coalgebraic approach based on paths as outlined in the introduction.

▶ **Definition 12.** A *labelled transition system* is a triple $(X, A_\tau, \to)$, where X is a set of states, A_τ a set of actions, and $\to \subseteq X \times A_\tau \times X$ is the so-called *transition relation*.

As usual, we write $x \xrightarrow{a} x'$ and $\twoheadrightarrow \; \subseteq X \times A^* \times X$ to denote an element $(x, a, x') \in \to$ and the weak reachability relation, respectively. The latter is defined as the smallest relation satisfying the following inference rules: $\dfrac{}{x \overset{\varepsilon}{\twoheadrightarrow} x}$ $\qquad$ $\dfrac{x \overset{\sigma}{\twoheadrightarrow} x' \;\; x' \xrightarrow{a} x''}{x \overset{\sigma a}{\twoheadrightarrow} x''}$.

▶ **Definition 13.** Let $(X, A_\tau, \to)$ be a labelled transition system. A symmetric relation $R \subseteq X \times X$ is called a *branching bisimulation* relation [25] if and only if for any $x, y, x' \in X$ and $a \in A_\tau$, if $x \xrightarrow{a} x' \wedge xRy$ then $(x'Ry \wedge a = \tau) \vee \exists_{y', y''}\ y \xrightarrow{\varepsilon} y' \xrightarrow{a} y'' \wedge xRy' \wedge x'Ry''$. Two states $x \in X$ and $x' \in X$ are *branching bisimlar* if and only if there exists a branching bisimulation relation R such that xRx'.

Next, we construct a $\mathcal{P} \circ \mathrm{Path}_\sim$-coalgebra based on paths, where $\mathcal{P}$ is the covariant power set endofunctor on the category of sets **Set**.

An *execution* starting from a state $x \in X$ of a labelled transition system $(X, A_\tau, \to)$ is a path $p \in \mathrm{Path}(X)$ such that $p(\sigma) \xrightarrow{a} p(\sigma a)$, for all $\sigma a \in \mathrm{dom}(p)$. Let $\mathrm{Exec}(x)$ be the set of all executions starting from x. Such a transition system can be modelled as a coalgebra $(X, \pi_X \circ \alpha)$, where transition function α is given as:

$$\alpha(x) = \{p \mid p \in \mathrm{Exec}(x) \wedge \mathrm{trace}(p) \in \tau^* a\} \cup \{p \mid p \in \mathrm{Exec}(x) \wedge \mathrm{trace}(p) \in \tau^*\}.$$

▶ Remark. At this stage, we would like to highlight the distinction between a path and an execution made in this paper. It should be noted that all executions of a system (under investigation) are paths; however, the converse may not be true. This is not unusual because after all the executions of a system are generated on the basis of how behaviour of the system is specified (for instance, by the transition relation in the case of labelled transition systems and by the transition function in the case of fully probabilistic system).

Next, we state the main result of this section.

▶ **Theorem 14.** *Let $(X, A_\tau, \to)$ be a labelled transition system and $(X, \pi_X \circ \alpha)$ be the corresponding $\mathcal{P} \circ \mathrm{Path}_\sim$-coalgebra. Then, two states $x, x' \in X$ are branching bisimilar if and only if the states x, x' are $\mathcal{P} \circ \mathrm{Path}_\sim$-behaviourally equivalent.*

Proof. $\boxed{\Rightarrow}$ Let $R \subseteq X \times X$ be the largest branching bisimulation on the given labelled transition system. Then, from [25] we know that R is an equivalence relation. So let $X \xrightarrow{f} X/R$ be the quotient map. Now to show that f is indeed the required $\mathcal{P} \circ \mathrm{Path}_\sim$-coalgebra homomorphism, we first construct a coalgebra $X/R \xrightarrow{\beta} \mathcal{P}\mathrm{Path}_\sim(X/R)$:

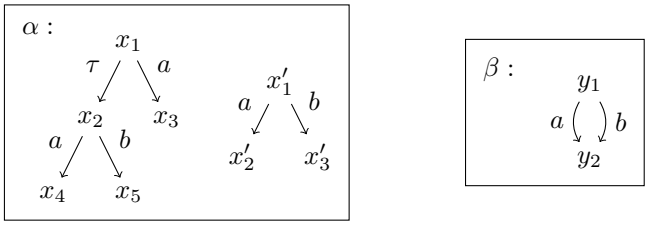
$$\beta(f(x)) = \{\mathrm{Path}_\sim(f)(p) \mid p \in \alpha(x)\}, \qquad \text{for all } x \in X\ .$$

Clearly, β is a total function because f is surjective. Next, we claim that β is well-defined, i.e., independent of the chosen representative. Let $x, x' \in X$ such that $f(x) = f(x')$. Then, we need to show that $\beta(f(x)) = \beta(f(x'))$. Suppose $[fp]_\sim \in \beta(f(x))$ with $p \in \alpha(x)$. Then, by structural induction on the word $\sigma \in \mathrm{dom}(p)$ we show that there is a path $p' \in \alpha(x')$ such that $f \circ (p|_\sigma) \sim f \circ p'$. Here, we write $p|_\sigma$ to denote the restriction of the function p to the sub-domain $\downarrow\sigma$. To see this, without loss of generality, let $\sigma a \in \mathrm{dom}(p)$. Then by the induction hypothesis we find an execution $p' \in \alpha(x')$ such that $f \circ (p|_\sigma) \sim f \circ p'$. Note that $p(\sigma) \xrightarrow{a} p(\sigma a)$ and using Lemma 9(3) we get $f \circ (p|_\sigma) \sim f \circ p' \implies p(\sigma)\ R\ \mathrm{last}(p')$. Let $a \in A$. Then, using the transfer property of branching bisimulation we get $\mathrm{last}(p') \xrightarrow{\varepsilon} y \xrightarrow{a} y'$ such that $p(\sigma)Ry$ and $p(\sigma a)Ry'$ since $p(\sigma)$ and $\mathrm{last}(p')$ are branching bisimilar. Moreover, from the stuttering lemma [25] we know that any intermediate state visited in the path $\mathrm{last}(p') \xrightarrow{\varepsilon} y$ is also R-related to $p(\sigma)$. Therefore, there is a path $p'' \in \mathrm{Path}(X)$ which extends p' such that $f \circ (p|_{\sigma a}) \sim f \circ p''$. In addition, if $a = \tau$ then we either have $p(\sigma\tau)\ R\ \mathrm{last}(p')$ or $\mathrm{last}(p') \xrightarrow{\varepsilon} y \xrightarrow{\tau} y'$, for some y, y', with $p(\sigma)Ry$ and $p(\sigma\tau)Ry'$. Suppose the former is true, then clearly we have $f \circ (p|_{\sigma\tau}) \sim f \circ p'$. The latter case is similar to the case when $a \in A$. Thus, for every $p \in \alpha(x)$ there is a path $p' \in \alpha(x')$ such that $f \circ p \sim f \circ p'$. Likewise, we can

show the symmetric property when the role of x and x' is interchanged. This completes the proof of the above claim. Clearly, we have $\beta \circ f = \mathcal{P}\mathrm{Path}_\sim(f) \circ \alpha$.

$\boxed{\Leftarrow}$ Let (Y, β) be a $\mathcal{P} \circ \mathrm{Path}_\sim$-coalgebra and $X \xrightarrow{f} Y$ be a $\mathcal{P} \circ \mathrm{Path}_\sim$-coalgebra homomorphism. Below we rather illustrate why the relation $xRx' \iff f(x) = f(x')$ is a witnessing branching bisimulation. The complete proof can be found in [6]. ◀

Consider the two labelled transition systems drawn below enclosed inside the two rectangles. Here, $X \xrightarrow{\alpha} \mathcal{P}\mathrm{Path}_\sim(X)$ and $Y \xrightarrow{\beta} \mathcal{P}\mathrm{Path}_\sim(Y)$ denote the corresponding path-based



coalgebras with $X = \{x_i, x'_j \mid i \in \{1,2,3,4,5\}, j \in \{1,2,3\}\}$ and $Y = \{y_1, y_2\}$. Furthermore, let $X \xrightarrow{f} Y$ be a function defined as $f(x) = y_1$ if $x \in \{x_1, x'_1, x_2\}$; otherwise $f(x) = y_2$. To illustrate why R (as defined above) is a witnessing branching bisimulation, consider the transition $x'_1 \xrightarrow{b} x'_3$ and $x_1 R x'_1$. Clearly, $\langle x'_1\ b\ x'_3\rangle \in \alpha(x_1)$, which further implies that $\langle y_1\ b\ y_2\rangle \in \beta(y_1)$. Since $\mathcal{P}\mathrm{Path}_\sim(f) \circ \alpha = \beta \circ f$ we know that there is an execution p such that $f \circ p$ is stutter equivalent to $\langle y_1\ b\ y_2\rangle$. And by inspection we note that $p = \langle x_1\ \tau\ x_2\ b\ x_5\rangle$ is such an execution. Moreover, $x_2 R y_1$ and $x_5 R y_2$ which is required by the transfer property of a branching bisimulation relation.

In hindsight, using the terminology of [24], a $\mathcal{P}\mathrm{Path}_\sim$-coalgebra homomorphism preserves the branching structure of states. As a consequence, two behaviourally equivalent states have the same set of executions under the image of a $\mathcal{P}\mathrm{Path}_\sim$-coalgebra homomorphism up to stutter invariance. For instance, in the above example, the sets of all executions having trace $\tau^* a$ from the states x_1 and x'_1 are $\{\langle x_1\ a\ x_3\rangle, \langle x_1\ \tau\ x_2\ a\ x_4\rangle\}$ and $\{\langle x'_1\ a\ x'_2\rangle\}$, respectively. Notice that the f-image of these two sets are equivalent up to stutter invariance. A similar argument can be observed for the set of executions from x_1, x'_1 having trace $\tau^* b$.

Though we have focussed on branching bisimulation, this approach can also be used to capture weak, η and delay bisimulation, by defining α differently, saturating τ leading transitions, trailing τ transitions or both, respectively. This is made explicit in [6].

4 A measurable space on paths

As mentioned in the introduction, we will consider coalgebras of type $\mathcal{G} \circ \mathrm{Path}_\sim$ over the base category **Meas** to characterise probabilistic delay bisimulation. However, before we do so, we have to fix which subsets of the sets $\mathrm{Path}(X)$ and $\mathrm{Path}_\sim(X)$ are measurable together with the construction of a measure on the space of paths, which can be a challenging issue in its own right. In this section, we resolve these fundamental issues by first recalling some basic definitions of measure theory taken from [17].

▶ **Definition 15.** A set $\Sigma_X \subseteq \mathcal{P}(X)$ of subsets of X is a *sigma-algebra* on X if and only if $X \in \Sigma_X$ and Σ_X is closed under the set complements and countable unions. Then, the tuple (X, Σ_X) is called a *measurable space*. A *measure* space is a measurable space (X, Σ_X) with a

measure $\Sigma_X \xrightarrow{\mu_X} [0,\infty]$, i.e., μ_X is a function satisfying $\mu(\emptyset) = 0$ and the *sigma-additivity* property: for any countable family of pairwise disjoint sets $U_i \in \Sigma_X$ (for $i \in I$) we have

$$\mu_X(\bigcup_{i\in I} U_i) = \sum_{i\in I} \mu_X(U_i) \ .$$

A *probability* space (X, Σ_X, μ_X) is a measure space with $\mu_X(X) = 1$. A *discrete* space is a measure space such that X is countable and $\Sigma_X = \mathcal{P}(X)$.

Here, the arbitrary sum of a family $\{r_i \mid i \in I\}$ of nonnegative real numbers is defined as $\sum_{i\in I} r_i = \sup\{\sum_{i\in J} r_i \mid J \subseteq_{\mathrm{f}} I\}$ (cf. [22]), where $J \subseteq_{\mathrm{f}} I \iff J \subseteq I \wedge J$ is a finite set.

We want to endow a notion of measurability on the set $\mathrm{Path}_\sim(X)$; however, for simplicity we first restrict ourselves to the set of all paths on X, i.e., $\mathrm{Path}(X)$. It turns out that the set of all paths carries a topological structure (precisely, they form what is known as Alexandroff topology [2]) and also satisfies the so-called Kolmogorov separability axiom. Once we have a topological space, the convention is to consider the smallest sigma-algebra generated by the set of all open sets (also known as the Borel sigma-algebra) as the set of measurable sets.

▶ **Definition 16.** A *topology* on a set X consists of a set of open sets $\mathcal{O}_X \subseteq \mathcal{P}(X)$ such that: first, the empty set and the whole space are in $\mathcal{O}_X$; second, the set $\mathcal{O}_X$ is closed under finite intersection and arbitrary unions. A topological space $(X, \mathcal{O}_X)$ is an *Alexandroff* space if the set $\mathcal{O}_X$ is closed under arbitrary intersection. A topological space $(X, \mathcal{O}_X)$ satisfies the *Kolmogorov separability axiom* (X is a T_0 space) if any two distinct points are topologically distinguishable, i.e., $\forall_{x,x'\in X}\ x \neq x' \implies \exists_{U\in\mathcal{O}_X}\ (x \in U \wedge x' \notin U) \vee (x \notin U \wedge x' \in U)$.

It is well-known that the set of all upward closed subsets generated by a poset forms a T_0 Alexandroff space. In particular, our set of paths $\mathrm{Path}(X)$ carries the following order:

$$p \preceq q \iff \mathrm{dom}(p) \subseteq \mathrm{dom}(q) \wedge \forall_{\sigma\in\mathrm{dom}(p)}\ q(\sigma) = p(\sigma) \ .$$

Actually, the above ordering is a prefix order in the sense of Cuijpers [11].

▶ **Definition 17.** A *prefix order* is a partial order whose every principal ideal is a totally ordered set.

▶ **Proposition 18.** *The history of a path $p \in \mathrm{Path}(X)$ is downward total, i.e., the set $\downarrow p = \{p' \mid p' \preceq p\}$ is a totally ordered set.*

▶ **Proposition 19.** *The set of all paths $\mathrm{Path}(X)$ on a set X forms a T_0 Alexandroff space, whose open sets are upward closed subsets of $\mathrm{Path}(X)$, i.e., $\mathcal{O}_{\mathrm{Path}(X)} = \{U \subseteq \mathrm{Path}(X) \mid U = \uparrow U\}$. Here, the set $\uparrow U = \{p' \mid \exists\ p \in U \wedge p \preceq p'\}$ denotes future of paths in the set U.*

At this stage, we note the following relationship between stutter paths and the order $\preceq$.

▶ **Lemma 20.** *Let X be a set. Then we have the following property: for any two paths $p_1, p_2 \in \mathrm{Path}(X)$, if $p_1^\dagger \preceq p_2^\dagger$ then $\exists_{p\in\mathrm{Path}(X)}\ p \sim p_1 \wedge p \preceq p_2$.*

Every point $x \in X$ in an Alexandroff space has a special neighbourhood associated with it, often called the *smallest neighbourhood* of x, denoted $\mathcal{N}(x) = \bigcap\{U \mid U \in \mathcal{O}_X \wedge x \in U\}$. In particular, this structure, in the case of paths, is the principal filter generated by a path.

▶ **Proposition 21.** *For a path $p \in \mathrm{Path}(X)$, the smallest neighbourhood of p represents the future of the path p, i.e., $\uparrow p = \mathcal{N}(p)$. In contrast, the closure $\mathrm{cl}(p)$ of a path $p \in \mathrm{Path}(X)$ – the smallest closed set that contains p – represents the history of p, i.e., $\mathrm{cl}(p) = \downarrow p$.*

The next proposition states that the subsets of paths which belong to the Borel sigma-algebra $\mathcal{B}(\mathcal{O}_{\mathrm{Path}(X)})$ are measurable.

▶ **Proposition 22.** *The tuple* $(\mathrm{Path}(X), \Sigma_{\mathrm{Path}(X)})$ *is a measurable space, where* $\Sigma_{\mathrm{Path}(X)} = \mathcal{B}(\mathcal{O}_{\mathrm{Path}(X)})$*. Here,* $\mathcal{B}(\mathcal{X})$ *denotes the smallest sigma-algebra generated by* $\mathcal{X} \subseteq \mathcal{P}(X)$*.*

Next, we establish that the $\mathrm{Path}(X) \xrightarrow{\mathrm{Path}(f)} \mathrm{Path}(Y)$ (for a given $X \xrightarrow{f} Y$) is measurable, i.e., if $V \in \Sigma_{\mathrm{Path}(Y)}$ then $f^{-1}V \in \Sigma_{\mathrm{Path}(X)}$. For this, we need the following result.

▶ **Theorem 23.** *For any* $X \xrightarrow{f} Y$*, the function* $\mathrm{Path}(f)$ *is an order embedding, i.e., for any* $p, p' \in \mathrm{Path}(X)$ *we have* $p \preceq p' \iff f \circ p \preceq f \circ p'$*.*

Since every order preserving function is continuous and every continuous function is Borel measurable, it follows that, in particular, $\mathrm{Path}(f)$ is Borel measurable.

▶ **Corollary 24.** *For any* $X \xrightarrow{f} Y$*, the function* $\mathrm{Path}(f)$ *is measurable.*

In hindsight, the function $\mathrm{Path}(f)$ is an arrow in the category **Meas**.

Next, we turn our attention on constructing a measurable space on the set $\mathrm{Path}_\sim(X)$. The idea is to first define an order on the quotient space $\mathrm{Path}_\sim(X)$, which can be inherited from the underlying space of paths $\mathrm{Path}(X)$ by simply letting: $[p]_\sim \preceq [q]_\sim \iff p^\dagger \preceq q^\dagger$, for all $p, q \in \mathrm{Path}(X)$.

▶ **Lemma 25.** *The relation* $\preceq$ *on the set* $\mathrm{Path}_\sim(X)$ *is a well-defined partial order. Furthermore, the relation* $\preceq$ *on the set* $\mathrm{Path}_\sim(X)$ *is also a prefix order.*

Once we have an order on the quotient space, we can establish that the quotient maps are order preserving (or continuous in the topological sense).

▶ **Theorem 26.** *The quotient function* $\mathrm{Path}(X) \xrightarrow{\pi} \mathrm{Path}_\sim(X)$ *is order preserving. Consequently, the quotient function* $\mathrm{Path}(X) \xrightarrow{\pi} \mathrm{Path}_\sim(X)$ *is Borel measurable, where the sigma-algebra on paths is given by* $\Sigma_{\mathrm{Path}_\sim(X)} = \mathcal{B}(\mathcal{O}_{\mathrm{Path}_\sim(X)})$*.*

Next, we state the main theorem of this section.

▶ **Theorem 27.** *For any* $X \xrightarrow{f} Y$*, the function* $\mathrm{Path}_\sim(f)$ *is order preserving. Thus, the function* $\mathrm{Path}_\sim(f)$ *is Borel measurable.*

Constructing measures on the space of paths

Often, measures on a space are constructed in a top-down manner by identifying a measurable set of building blocks and defining a set-function on this collection (for example, in the case of Lebesgue measures on $\mathbb{R}$, a semi-closed interval $[r, r')$ with $r \leq r'$ is one such building block and the set-function maps every interval of the form $[r, r')$ to the value $r' - r$). In turn, measure extension theorems (for instance, the well-known Carathéodory-Hahn extension theorem; see [18, pp 356]) are invoked to lift the set-function on building blocks to a measure on the whole measurable space. In this paper, we will follow a similar recipe; our building blocks will be open subsets of paths. As for measure extension theorems, we will use a result (cf. Theorem 31) established by Alvarez-Manilla [3]. Below, we recall some definitions on a topological space necessary to state this result.

▶ **Definition 28.** Let $(X, \mathcal{O}_X)$ be a topological space. A function $\mathcal{O}_X \xrightarrow{\mu} [0,\infty]$ is a *valuation* if and only if the following conditions are satisfied.

1. The function μ is strict, i.e., $\mu(\emptyset) = 0$
2. The function μ is order preserving, i.e., for any two open sets $U, U' \in \mathcal{O}_X$, we have $U \subseteq U'$ implies $\mu(U) \leq \mu(U')$.
3. The function μ is modular, i.e., for any two open sets $U, U' \in \mathcal{O}_X$, we have $\mu(U)+\mu(U') = \mu(U \cup U') + \mu(U \cap U')$.

A valuation μ is *Scott-continuous* if and only if for any directed family of open sets $(U_i)_{i\in I}$ we have $\mu(\bigcup_{i\in I} U_i) = \sup_{i\in I} \mu(U_i)$. Lastly, a valuation μ is *locally finite* if and only if every point has a finitely valued open neighbourhood.

▶ **Definition 29.** A space $(X, \mathcal{O}_X)$ is *locally compact* if and only if for every point x and open set U with $x \in U$, there is a compact subset $V \subseteq X$ such that $x \in \text{int}(V)$ and $V \subseteq U$. Here, $\text{int}(V)$ denotes the interior of $V \subseteq X$.

▶ **Definition 30.** A topological space $(X, \mathcal{O}_X)$ is *sober* if and only if every irreducible closed set is a closure of a unique point. A closed set C is *irreducible* if and only if C is nonempty and it cannot be expressed as union of two smaller closed subsets, i.e., if $C = C_1 \cup C_2$ and C_1, C_2 are closed sets, then $C = C_1$ or $C = C_2$.

We call a subset $C \subseteq X$ *non-sober* if C is irreducible, C is closed, and it cannot be stated as a closure of point (i.e., $\nexists_{x\in X}\ C = \text{cl}(x)$).

▶ **Theorem 31** ([3])**.** *Every locally finite and Scott-continuous valuation on a locally compact sober space extends uniquely to a Borel measure.*

The restrictions on $\mathcal{O}_X \xrightarrow{\mu} [0,\infty]$ imposed by the above theorem are not unreasonable; at-least for our purpose. In Section 5, we will construct a locally finite and a Scott-continuous valuation on open subsets of paths, which is induced by a given fully-probabilistic transition system. Nevertheless, we cannot immediately apply Theorem 31 because our space $\text{Path}(X)$ is not a sober space, even though it is locally compact, i.e., every path $p \in \text{Path}(X)$ has a compact neighbourhood (since $p \in \uparrow p$). As a result, in the following, we first 'soberify' our space $\text{Path}(X)$ and use Theorem 31 to construct a Borel measure on $\text{Path}(X)$ by lifting a given locally finite and Scott-continuous valuation $\mathcal{O}_{\text{Path}(X)} \xrightarrow{\mu} [0,\infty]$.

▶ Remark. By inspection, we note that our space $\text{Path}(X)$ is non-sober. For instance, if X is non-empty then unfolding a τ-loop results in an infinite chain of paths without any maximum since the domain of a path is a set of prefixes generated by some *finite* word.

Recall that, for a set X, both sets of paths $\text{Path}(X)$ and stutter-equivalent paths $\text{Path}_\sim(X)$ are prefix orders. We want to construct measures on both kinds of spaces, therefore below we work with a class of *simple* prefix orders which generalises both the structures.

▶ **Definition 32.** A prefix order is *simple* if the history of every point is a finite set.

For example, the sets $\text{Path}(X)$ and $\text{Path}_\sim(X)$ are simple prefix orders.

▶ **Proposition 33.** *A directed subset of a prefix order is always totally ordered. In addition, an irreducible downward closed subset of a prefix order is always totally ordered.*

Next, we construct a space X^∞ consisting of all points from X in which the non-sober sets (w.r.t. Alexandroff topology) are added as limit points.

$$X^\infty = X \cup \{\infty_C \mid C \subseteq X \text{ is a non-sober set w.r.t. Alexandroff topology}\}.$$
$$\preceq' = \preceq \cup \{(\infty_C, \infty_C) \mid \infty_C \in X^\infty\} \cup \{(x, \infty_C) \mid x \in C\}.$$

As an example, consider the prefix order $(\mathbb{N}, \leq)$ with their natural ordering. The sober space $\mathbb{N}^\infty = \mathbb{N} \cup \{\infty_\mathbb{N}\}$ is isomorphic to the well-known set of extended natural numbers $\mathbb{N}_\omega$.

▶ **Lemma 34.** *The set X^∞ is prefix ordered by the relation $\preceq'$, if $(X, \preceq)$ is a prefix order.*

Henceforth, we do not distinguish between the relation $\preceq$ and $\preceq'$. Notice that being sober is a topological property and therefore, we need a 'right' notion of topology on X^∞ to qualify it as sober. For instance, if we take upward closed sets as open sets (just like in the case of X) we find that the space X^∞ is still non-sober; as a result, X^∞ is non-sober w.r.t. Alexandroff topology. However, if we endow X^∞ with a Scott topology then the space becomes sober w.r.t. this finer topology. For example, in the case of extended natural numbers, the problematic case of the directed set $\mathbb{N}$ (which was non-sober w.r.t. Alexandroff topology) is actually not a Scott-closed set[2] since $\sup \mathbb{N} = \infty_\mathbb{N}$ and $\infty_\mathbb{N} \notin \mathbb{N}$.

▶ **Proposition 35.** *Let $(X, \preceq)$ be a simple prefix order. A subset $U \subseteq X^\infty$ is* Scott *open if and only if U is upward closed and it is inaccessible by directed joins, i.e., for any directed set $D \subseteq X^\infty$ if $\sup D$ exists and $\sup D \in U$ then $D \cap U \neq \emptyset$. Let $\mathcal{S}_{X^\infty}$ denote the collection of Scott open subsets of X^∞. Then, the space $(X^\infty, \mathcal{S}_{X^\infty})$ is a sober space.*

To apply Theorem 31, we need to first construct a locally finite and Scott-continuous valuation on our new sober space $\mathrm{Path}^\infty(X)$. In the following theorem, we will construct one such valuation on $\mathrm{Path}^\infty(X)$ from an old valuation $\mathcal{O}_{\mathrm{Path}(X)} \xrightarrow{\mu} [0, \infty]$.

▶ **Theorem 36.** *Let $(X, \preceq)$ be a prefix order and let $\mathcal{O}_X \xrightarrow{\mu} [0, \infty]$ be a locally finite and Scott-continuous valuation. Then, the function $\mathcal{S}_{X^\infty} \xrightarrow{\tilde{\mu}} [0, \infty]$ defined as follows:*

$$\tilde{\mu}(V) = \mu(V \cap X) \quad \textit{(for every Scott-open set } V \in \mathcal{S}_{X^\infty}\textit{)}$$

is a Scott-continuous valuation. If X is simple then $\tilde{\mu}$ is locally finite.

As a result, the function $\tilde{\mu}$ lifts to a unique Borel measure on the sigma-algebra $\Sigma_{\mathrm{Path}^\infty(X)} = \mathcal{B}(\mathcal{S}_{\mathrm{Path}^\infty(X)})$ due to Theorem 31. However, in order to reflect back this measure on the original sigma-algebra $\Sigma_{\mathrm{Path}(X)}$, it is sufficient to establish that $\Sigma_{\mathrm{Path}(X)}$ is contained in the Borel sigma-algebra induced by Scott-open sets, i.e., $\Sigma_{\mathrm{Path}(X)} \subseteq \Sigma_{\mathrm{Path}^\infty(X)} = \mathcal{B}(\mathcal{S}_{\mathrm{Path}^\infty(X)})$. The next theorem states under what conditions the set-containment between the two sigma-algebras $\Sigma_{\mathrm{Path}(X)}, \Sigma_{\mathrm{Path}^\infty(X)}$ is possible.

▶ **Theorem 37.** *For countable sets A and X, the sigma-algebra $\Sigma_{\mathrm{Path}(X)}$ is contained in the Borel sigma-algebra $\Sigma_{\mathrm{Path}^\infty(X)}$. Moreover, we also have $\Sigma_{\mathrm{Path}_\sim(X)} \subseteq \Sigma_{\mathrm{Path}^\infty_\sim(X)}$.*

▶ **Corollary 38.** *Suppose the sets A and X are countable. Then every locally finite and Scott-continuous valuations $\mathcal{O}_{\mathrm{Path}(X)} \xrightarrow{\mu} [0, \infty]$ and $\mathcal{O}_{\mathrm{Path}_\sim(X)} \xrightarrow{\mu} [0, \infty]$ lifts to a unique Borel measure $\Sigma_{\mathrm{Path}(X)} \xrightarrow{\tilde{\mu}} [0, \infty]$ and $\Sigma_{\mathrm{Path}_\sim(X)} \xrightarrow{\tilde{\mu}} [0, \infty]$, respectively.*

5 Probabilistic delay bisimulation

In this section, we use the concepts developed in the previous section to characterise probabilistic delay bisimulation relation between the states of a fully probabilistic system.

[2] A subset $C \subseteq X$ of a prefix order $(X, \preceq)$ is Scott closed if and only if C is downward closed and for any directed set $D \subseteq C$, if $\sup D$ exists then $\sup D \in C$.

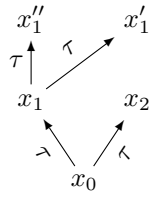


Figure 3 An example motivating separation closure.

▶ **Definition 39.** A *(fully) probabilistic transition system* is a triple (X, A_τ, P) consisting of a countable set of states X, a countable set of actions A_τ, and a *probability transition function* $X \times A_\tau \times X \xrightarrow{P} [0,1]$ such that for every $x \in X$, the set $\{(a, x') \mid 0 < P(x, a, x')\}$ is finite and $\sum_{(a,x') \in A_\tau \times X} P(x, a, x') \in \{0, 1\}$.

Given a probabilistic transition system (X, A_τ, P), an *execution* p is a path on X such that $\forall_{\sigma a \in \mathrm{dom}(p)}\ 0 < P(p(\sigma), a, p(\sigma a))$. Let $\mathrm{Exec}(x)$ be the set of all executions starting from the state x. We write $\hat{a} = \varepsilon$ if $a = \tau$ and $\hat{a} = a$ if $a \in A$.

▶ **Definition 40.** An equivalence relation $R \subseteq X \times X$ on a probabilistic transition system (X, A_τ, P) is a *probabilistic delay bisimulation* [4, 22] if and only if

$$\forall_{x,x' \in X}\ xRx' \implies \forall_{x'' \in X, a \in A_\tau}\ \ P(x, \tau^* \hat{a}, [x'']_R) = P(x', \tau^* \hat{a}, [x'']_R)\ .$$

Two states $x, x' \in X$ are *probabilistic delay bisimilar* if and only if there is a probabilistic delay bisimulation R such that xRx'.

Here, the probabilities associated with weak transitions are defined (originally given in [21]) in the following way. For $x \in X, Y \subseteq X, L \subseteq A_\tau^*$, we let

- $x \xrightarrow{L} Y = \{p \in \mathrm{Exec}(x) \mid \mathrm{trace}(p) \in L \wedge \mathrm{last}(p) \in Y \ \wedge$
  $\forall_q\ (q \prec p \wedge \mathrm{trace}(q) \in L) \implies \mathrm{last}(q) \notin Y\}$.
- $P(x, L, Y) = \sum_{p \in x \xrightarrow{L} Y} \mu_P(p)$, where $\mathrm{Path}(X) \xrightarrow{\mu_P} [0,1]$ is defined as:

$$\mu_P(p) = \begin{cases} \prod_{\sigma a \in \mathrm{dom}(p)} P(p(\sigma), a, p(\sigma a)), & \text{if } p \in \mathrm{Exec}(x), \text{ for some } x \in X \\ 0, & \text{otherwise} \end{cases}.$$

▶ **Proposition 41.** *For a given fully probabilistic system (X, A_τ, P), the induced function μ_P on paths is order reversing. Moreover, $\mu_P(\varepsilon_x) = 1$ (for any $x \in X$).*

In contrast to Section 3, our base category will be rather the category of measurable spaces and measurable functions **Meas**.

▶ **Definition 42.** Below we recall the well known Giry functor $\mathcal{G}$ (see e.g. [17]):

- Let (X, Σ_X) be a measurable space. Then, $\mathcal{G}(X, \Sigma_X) = (\mathcal{G}X, \Sigma_{\mathcal{G}X})$, where $\mathcal{G}X$ is the set of all probability measures on the measurable space (X, Σ_X). The sigma-algebra $\Sigma_{\mathcal{G}X}$ is the smallest sigma-algebra such that the evaluation maps $\mathcal{G}X \xrightarrow{\epsilon_U} [0,1]$ are Borel measurable, for every $U \in \Sigma_X$.
- For any arrow $X \xrightarrow{f} Y$ in **Meas**, we let $\mathcal{G}(f)(\mu) = \mu \circ f^{-1}$.

To motivate the next definition, consider the transition system depicted in Figure 3 and assume a nonzero probability with each of the drawn transitions. Furthermore, let p_1, p_1', and p_2 be the three executions that reach the states x_1, x_1', and x_2, resp., from the state x_0. Notice that $P(x_0, \tau^*, \{x_1, x_1', x_2\})$ is the sum of the probabilities associated only with the executions p_1, p_2. The execution p_1' is not considered in the above computation because one can reach the set of target states $\{x_1, x_1', x_2\}$ with the execution p_1 which is a prefix of p_1'. This means, the execution p_1' is redundant and neglected while computing the probability to reach the above target states. Such redundancies at the level of paths are identified by the following notion of separation closure.

▶ **Definition 43.** Let $(X, \mathcal{O}_X)$ be an Alexandroff space. The *separation closure* of a subset $U \subseteq X$ is the set $U^\star = \{x \in U \mid \mathrm{cl}(x) \cap U = \{x\}\}$. A subset $U \subseteq X$ is *separated* if $U = U^\star$.

In the context of the previous example (Figure 3), let $U = \{p_1, p_1', p_2\}$. Then, we find that $\downarrow x \cap U = \{x\}$, for $x \in \{p_1, p_2\}$, while for the execution p_1' we find that $\downarrow p_1' \cap U = \{p_1, p_2\}$. Thus, $U^\star = \{p_1, p_2\}$ which were the only executions needed to compute the probability to reach one of the target states. Incidentally, a separated subset of paths $U \subseteq \mathrm{Path}(X)$ (i.e., $U = U^\star$) is minimal in the sense that any two distinct paths $p, p' \in U$ are not in the prefix relation $\preceq$, i.e., $p \not\preceq p'$ and $p' \not\preceq p$. The following proposition asserts this.

▶ **Proposition 44.** *In an Alexandroff space $(X, \mathcal{O}_X)$, a separated subset $U \subseteq X$ (i.e., $U = U^\star$) has topologically distinguishable points. Moreover,*

1. *the separation closure of a set is always separated, i.e., $U^\star = U^{\star\star}$, for any $U \subseteq X$.*
2. *the collection of separated sets is hereditary, i.e., if $U_1 \subseteq U_2$ and $U_2 = U_2^\star$, then $U_1 = U_1^\star$.*
3. *for any subset $U \subseteq X$, we have $(\uparrow U)^\star \subseteq U^\star$. Moreover the converse also holds, if the underlying space X is a T_0 space. Here, upward closure is w.r.t. the specialisation order $\preceq$, i.e., $x \preceq x' \iff \mathrm{cl}(x) \subseteq \mathrm{cl}(x')$, for any $x, x' \in X$.*

Thus, separation closure provides an alternative way to compute $P(x, L, Y)$.

▶ **Lemma 45.** *For a given system (X, A_τ, P), define the set $x \overset{L}{\twoheadrightarrow} Y = \{p \in \mathrm{Path}(X) \mid p(\varepsilon) = x \wedge \mathrm{trace}(p) \in L \wedge \mathrm{last}(p) \in Y\}$. Then, $\sum_{p \in x \overset{L}{\longrightarrow} Y} \mu_P(p) = \sum_{p \in (x \overset{L}{\twoheadrightarrow} Y)^\star} \mu_P(p)$.*

It should be noted that for a given probabilistic transition system (X, A_τ, P), we have $x \overset{L}{\longrightarrow} Y \subseteq (x \overset{L}{\twoheadrightarrow} Y)^\star$; however, the converse is not true in general.

The next theorem highlights a property characteristic to fully probabilistic systems. It states that if a separated subset of paths U has a lower bound p, i.e., $\forall_{q \in U}\ p \preceq q$, then the sum of probabilities associated with each path in U is bounded by the weight of p. This property is due to the order reversing nature of the function μ_P (cf. Proposition 41).

▶ **Theorem 46.** *Given a system (X, A_τ, P), a path $p \in \mathrm{Path}(X)$, and a separated set of paths $U \subseteq \mathrm{Path}(X)$ such that $p \preceq U$, i.e., $\forall_{q \in U}\ p \preceq q$. Then, $\sum_{q \in U} \mu_P(q) \leq \mu_P(p)$.*

Next we focus on the construction of probability measures on the space $\mathrm{Path}(X)$. From Corollary 38, it suffices to construct a locally-finite and Scott-continuous valuation on the open subsets of paths. The following theorem extends the function μ_P (induced by a given fully probabilistic system (X, A_τ, P)) to such a valuation on paths.

▶ **Theorem 47.** *Given a system (X, A_τ, P), then the function $\mathcal{O}_{\mathrm{Path}(X)} \xrightarrow{\tilde{\mu}_P} [0, \infty]$ defined as $\tilde{\mu}_P(U) = \mu_P(U^\star)$ (for every open set U) is a locally finite and Scott-continuous valuation.*

Now we have all the technical machinery to encode a given probabilistic transition system (X, A_τ, P) as a coalgebra $X \xrightarrow{\alpha} \mathcal{G}\mathrm{Path}_\sim(X)$, where $\Sigma_X = \mathcal{P}(X)$ (since X is countable). The transition system α is defined, coalgebraically, as follows:

$$\alpha(x)(U) = \tilde{\mu}_P({\pi_X}^{-1}(U) \cap \uparrow\varepsilon_x), \qquad \text{for every } U \in \Sigma_{\mathrm{Path}_\sim(X)} \ . \tag{2}$$

Here, we abuse notation by using $\tilde{\mu}_P$ to denote a measure on $\Sigma_{\mathrm{Path}(X)}$. Note that this measure is rather constructed by extending the valuation given in Theorem 47.

▶ **Proposition 48.** *The mapping in* (2) *is a probability measure.*

Now we are ready to state the main result of this section.

▶ **Theorem 49.** *Two states are probabilistic delay bisimilar if and only if they are* $\mathcal{G}\circ\mathrm{Path}_\sim$*-behaviourally equivalent.*

Proof. $\boxed{\Rightarrow}$ Let R be a probabilistic delay bisimulation, let (X, α) be the $\mathcal{G}\circ\mathrm{Path}_\sim$-coalgebra induced by (X, A_τ, P), and let $X \xrightarrow{f} X/R$ be the quotient map. We will construct a coalgebra on the quotient set X/R in two stages. First, we construct a measure $\nu_{f(x)}$ (for each $x \in X$) on the space $\mathrm{Path}_\sim(X/R)$ using the extension result (cf. Corollary 38) such that it coincides with the pushforward measure $(\alpha(x))_*$ on the open subsets $V \in \mathcal{O}_{\mathrm{Path}_\sim(X/R)}$. Second, we invoke the well-known application (taken from [17, Proposition 2.10]) of Dynkin's $\lambda - \pi$ theorem to conclude that $\nu_{f(x)} = (\alpha(x))_*$.

Let $x \in X$. Define a function $\mathcal{O}_{\mathrm{Path}_\sim(X/R)} \xrightarrow{\nu_{f(x)}} [0, \infty]$ as follows:

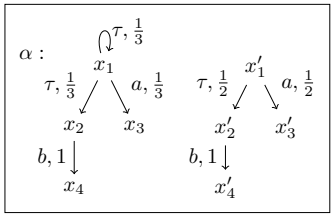
$$\nu_{f(x)}(V) = \tilde{\mu}_P(\pi^{-1}\mathrm{Path}_\sim(f)^{-1}V \cap \uparrow\varepsilon_x), \qquad \text{for every } V \in \mathcal{O}_{\mathrm{Path}_\sim(X/R)}. \tag{3}$$

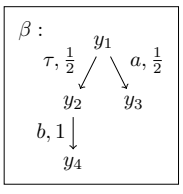
We need a technical result proven in [22, Lemma 24] in order to show that $\nu_{f(x)}$ is well defined, i.e., for any xRx' and open set $V \in \mathcal{O}_{\mathrm{Path}_\sim(X/R)}$ we have $\nu_{f(x)}(V) = \nu_{f(x')}(V)$. (See [6] for the proof of this claim). Moreover, the function $\nu_{f(x)}$ is a valuation, which immediately follows from (3) and the fact that μ is a valuation. Therefore, from Corollary 38, the valuation $\nu_{f(x)}$ extends to a Borel measure $\tilde{\nu}_{f(x)}$ on the space $\mathrm{Path}_\sim(X/R)$.

Recall that the pushforward measure $(\alpha(x))_*(V) = \alpha(x)(\mathrm{Path}_\sim(f)^{-1}(V))$ (for each Borel set $V \in \Sigma_{\mathrm{Path}_\sim(X/R)}$), is also a measure on the space $\mathrm{Path}_\sim(X/R)$. Clearly, for any $V \in \mathcal{O}_{\mathrm{Path}_\sim(X/R)}$, we have $(\alpha(x))_*(V) = \nu_{f(x)}(V) = \tilde{\nu}_{f(x)}(V)$ due to Equation (2). Since both $\tilde{\nu}_{f(x)}$ and $(\alpha(x))_*$ are probability measures, so from [17, Proposition 2.10] we get $\tilde{\nu}_{f(x)} = (\alpha(x))_*$. Now letting $\beta(f(x)) = \tilde{\nu}_{f(x)}$, we find that f is a coalgebra homomorphism because $\beta(f(x))(V) = \tilde{\nu}_{f(x)}(V) = (\alpha(x))_*(V) = \alpha(x)(\mathrm{Path}_\sim(f)^{-1}V)$, for every $V \in \Sigma_{\mathrm{Path}_\sim(X/R)}$.

$\boxed{\Leftarrow}$ Let (X, A_τ, P) be a fully probabilistic system and (X, α) be the corresponding $\mathcal{G} \circ \mathrm{Path}_\sim$-coalgebra. Moreover, let (Y, β) be a $\mathcal{G} \circ \mathrm{Path}_\sim$-coalgebra and $X \xrightarrow{f} Y$ be a $\mathcal{G}\circ\mathrm{Path}_\sim$ coalgebra homomorphism. In [6], we show that the equivalence relation $xRx' \iff f(x) = f(x')$ is a probabilistic delay bisimulation. Below we rather illustrate why R is a witnessing probabilistic delay bisimulation relation. ◀

Consider the two probabilistic transition systems drawn below,





together with the path-based coalgebras $X \xrightarrow{\alpha} \mathcal{G}\mathrm{Path}_\sim(X)$ and $Y \xrightarrow{\beta} \mathcal{G}\mathrm{Path}_\sim(Y)$ where $X = \{x_i, x'_j \mid i,j \in \{1,2,3,4\}\}$ and $Y = \{y_i \mid i \in \{1,2,3,4\}\}$. Furthermore, let $X \xrightarrow{f} Y$ be a function defined as $f(z_i) = y_i$, where $z \in \{x, x'\}$ and $i \in \{1,2,3,4\}$. To see why the relation R (as defined above) is a witnessing bisimulation, consider the equation

$$\alpha(x_1)\Big(\bigcup_{p \in x_1 \xrightarrow{\tau^* b} [x_4]_R} \uparrow [p]_\sim\Big) = \alpha(x'_1)\Big(\bigcup_{p \in x'_1 \xrightarrow{\tau^* b} [x_4]_R} \uparrow [p]_\sim\Big), \tag{4}$$

which can be derived from the facts $x_1 R x'_1$ and $\beta \circ f = \mathcal{G}\mathrm{Path}_\sim(f) \circ \alpha$ (see [6] for the proof in the general case). The two terms in Equation 4 denote the probabilities of reaching the equivalence class $[x_4]_R$ from the states x_1 and x'_1. This can be seen, for instance, by deriving $\alpha(x_1)(\bigcup_{p \in x_1 \xrightarrow{\tau^* b} [x_4]_R} \uparrow [p]_\sim) = P(x_1, \tau^* b, [x_4]_R)$ using Equation 2, definition of $\tilde{\mu}_P$, Proposition 3, and Lemma 45. Moreover, the probability to reach the equivalence class $[x_4]_R$ from x_1, x'_1 is $\frac{1}{2}$ because $P(x_1, \tau^* b, [x_4]_R) = \sum_{i=1}^{\infty} (\frac{1}{3})^i = \frac{1}{2} = P(x'_1, \tau^* b, [x_4]_R)$.

6 Discussion and conclusion

The main message of this paper is that behavioural equivalence in a path-based coalgebra is sufficient to capture branching bisimulation. In particular, we considered coalgebras of type $F \circ \mathrm{Path}_\sim$ over a concrete category $\mathbf{C}$, where F is an endofunctor modelling the branching type of the system under investigation. We showed that behavioural equivalence when $F = \mathcal{P}$ and $\mathbf{C} = \mathbf{Set}$ coincides with the traditional branching bisimulation [25]. In a similar spirit, we also showed that behavioural equivalence when $F = \mathcal{G}$ and $\mathbf{C} = \mathbf{Meas}$ coincides with the probabilistic delay bisimulation [4, 22, 21].

Interestingly, in the case of labelled transition systems, we can use the final chain based algorithm presented in [1] to minimise the system with respect to branching bisimulation. The following prerequisites for this algorithm are satisfied in this context: first, a terminal object exists in **Set**; second, **Set** is equipped with a (epi,mono)-factorisation structure; third, the functor $\mathcal{P} \circ \mathrm{Path}_\sim$ preserves monomorphisms. However, for the probabilistic case, more research is required to find out whether the above conditions are valid or not.

In retrospect, our paper comes short in one regard when comparing with the recent works [8, 10, 12] on capturing weak bisimulation; namely, there is no abstract construction given to construct our path-based coalgebras from the system under study. In particular, we would like to construct a path-based coalgebra, for instance, $X \xrightarrow{\alpha'} F\mathrm{Path}_\sim(X)$ from a given coalgebra of type $X \xrightarrow{\alpha} F(A_\tau \times X)$.

In this regard, it might be interesting to extend the initial work of Jacobs and Sokolova [13]: Given a system $X \longrightarrow TFX$ over **Set** (T is a monad modelling the branching type and F is an endofunctor modelling the transition type), then the traces (executions) can be described as an arrow $X \longrightarrow I$ ($X \times I \longrightarrow I$) in the Kleisli category of T with I being the initial algebra of F. Note that this insight of [13] works under some technical requirements and it is unclear whether these requirements hold in a more general setting of **Meas**. This was already voiced by Kerstan and König [15] in conjunction with generic trace semantics for probabilistic systems that were modelled over the base category **Meas**.

Another way to generalise the result of this paper is to consider the executions of a system as first-class citizens from the onset. Such a venture is carried out by Cuijpers [11] under the banner of prefix orders. Prefix orders are partially ordered sets whose principal ideals are totally ordered sets. The homomorphisms on such structures are called history

preserving functions, those order preserving functions that preserve the principal ideals of the underlying ordered sets. Beohar and Cuijpers [5] extended the theory of open maps [14] to the concrete category setting to get a characterisation of traditional branching bisimulation. Therefore, it will be worthwhile to study whether the measure theoretic concepts proposed here can be lifted to the more general setting of prefix orders to capture probabilistic branching bisimulation. Lastly, it will be interesting to construe a notion of behavioural equivalence in the open map approach akin to the theory of coalgebras, where the notion of bisimulation is parametric to a functor modelling the branching type of system under study.

Acknowledgements. The authors thank the anonymous reviewers of the conference CALCO'17 for their feedbacks on an earlier draft of this paper. The authors also thank Barbara König for various discussions regarding this work and for her earlier comments which led to significant improvements of this paper. The authors would also like to thank Pieter Cuijpers for his various feedbacks and continuing support over the course of this work.

References

1 J. Adámek, F. Bonchi, M. Hülsbusch, B. König, S. Milius, and A. Silva. A coalgebraic perspective on minimization and determinization. In *Proc. of FOSSACS '12*, pages 58–73. Springer, 2012. LNCS/ARCoSS 7213.

2 P. Alexandroff. Diskrete räume. *Rec. Math. N.S.*, 2(44)(3):501–519, 1937.

3 M. Alvarez-Manilla. Extension of valuations on locally compact sober spaces. *Topology and its Applications*, 124(3):397 – 433, 2002.

4 C. Baier and H. Hermanns. Weak bisimulation for fully probabilistic processes. In *Proc. of CAV*, pages 119–130, London, UK, 1997. Springer.

5 H. Beohar and P.J.L. Cuijpers. Open maps in concrete categories and branching bisimulation for prefix orders. *ENTCS*, 319:51 – 66, 2015. doi:10.1016/j.entcs.2015.12.005.

6 H. Beohar and S. Küpper. On path-based coalgebras and weak notions of bisimulation. *ArXiv e-prints*, May 2017. URL: https://arxiv.org/abs/1705.08715.

7 F. Bonchi, S. Milius, A. Silva, and F. Zanasi. Killing epsilons with a dagger: A coalgebraic study of systems with algebraic label structure. *TCS*, 604:102 – 126, 2015. Coalgebraic Methods in Computer Science.

8 T. Brengos. *On Coalgebras with Internal Moves*, pages 75–97. Springer, 2014. doi:10.1007/978-3-662-44124-4_5.

9 T. Brengos. Weak bisimulation for coalgebras over order enriched monads. *Logical Methods in Computer Science*, 11(2), 2015. doi:10.2168/LMCS-11(2:14)2015.

10 T. Brengos, M. Miculan, and M. Peressotti. Behavioural equivalences for coalgebras with unobservable moves. *JLAP*, 84(6):826 – 852, 2015. doi:10.1016/j.jlamp.2015.09.002.

11 P.J.L. Cuijpers. Prefix orders as a general model of dynamics. In I. Mackie, M. Ayala-Rincón, and E. Bonelli, editors, *Proc. of Developments in Computation Models*, DCM'13, pages 25–29, Buenos Aires, Argentina, 2013.

12 S. Goncharov and D. Pattinson. *Coalgebraic Weak Bisimulation from Recursive Equations over Monads*, pages 196–207. Springer, 2014. doi:10.1007/978-3-662-43951-7_17.

13 B. Jacobs and A. Sokolova. *Traces, Executions and Schedulers, Coalgebraically*, pages 206–220. Springer, 2009. doi:10.1007/978-3-642-03741-2_15.

14 A. Joyal, M. Nielson, and G. Winskel. Bisimulation and open maps. In *Proc. of 8th Annual IEEE Symposium on Logic in Computer Science*, pages 418–427, 1993. doi:10.1109/LICS.1993.287566.

15 H. Kerstan and B. König. Coalgebraic trace semantics for continuous probabilistic transition systems. *Logical Methods in Computer Science*, 9(4), 2013. doi:10.2168/LMCS-9(4:16)2013.

16 R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, Berlin Heidelberg, 1980.

17 P. Panangaden. *Labelled Markov Processes*. Imperial College Press, London, 2009.

18 H. L. Royden and P. M. Fitzpatrick. *Real analysis*. Pearson, 4th edition edition, 2010.

19 J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *TCS*, 249(1):3 – 80, 2000. doi:10.1016/S0304-3975(00)00056-6.

20 A. Sokolova. Personal communication. 11 2016.

21 A. Sokolova, E. de Vink, and H. Woracek. Weak bisimulation for action-type coalgebras. *ENTCS*, 122:211 – 228, 2005. doi:10.1016/j.entcs.2004.06.050.

22 A. Sokolova, E. de Vink, and H. Woracek. Coalgebraic weak bisimulation for action-type systems. *SACS*, 19:93–144, 2009.

23 R.J. van Glabbeek. *The linear time — Branching time spectrum II*, pages 66–81. Springer, 1993. doi:10.1007/3-540-57208-2_6.

24 R.J. van Glabbeek. *What is Branching Time Semantics and Why to Use It?*, pages 469–479. World Scientific Publishing, River Edge, USA, 2001.

25 R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, 43(3):555–600, May 1996. doi:10.1145/233551.233556.

Parity Automata for Quantitative Linear Time Logics*

Corina Cîrstea[1], Shunsuke Shimizu[2], and Ichiro Hasuo[3]

1 **University of Southampton, UK**
cc2@ecs.soton.ac.uk
2 **National Institute of Informatics, Tokyo, Japan**
shunsuke@nii.ac.jp
3 **National Institute of Informatics, Tokyo, Japan**
hasuo@nii.ac.jp

Abstract

We initiate a study of automata-based model checking for previously proposed quantitative linear time logics interpreted over coalgebras. Our results include: (i) an automata-theoretic characterisation of the semantics of these logics, based on a notion of extent of a quantitative parity automaton, (ii) a study of the expressive power of Büchi variants of such automata, with implications on the expressiveness of fragments of the logics considered, and (iii) a naïve algorithm for computing extents, under additional assumptions on the domain of truth values.

1998 ACM Subject Classification F.1.1 Models of Computation, F.4.1 Modal Logic

Keywords and phrases coalgebra, quantitative logic, linear time logic, parity automaton

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.7

1 Introduction

Linear time logics such as LTL or the linear time μ-calculus (see e.g. [9]), originally interpreted over non-deterministic transition systems, have been adapted and used successfully in model checking various types of non-deterministic and probabilistic systems [3]. These logics share the same notion of linear time behaviour, but depending on the branching present in the models, have either a qualitative or a quantitative interpretation (with $\{0, 1\}$, resp. the unit interval as domains of truth values). Despite commonalities between the logics and their automata-based verification techniques, a uniform account of the connection between (quantitative) linear-time logics and (quantitative) automata over infinite structures is still missing. This would allow existing verification techniques to be transferred to new models, including weighted ones (for which linear time logics have already been studied [15]).

This paper initiates a general study of the connection between quantitative linear time logics and quantitative automata on infinite structures, grounded in coalgebraic modelling, by building on recent work on *maximal traces* [6], quantitative linear time logics for systems with branching [4, 5], and coalgebraic trace semantics for Büchi and parity automata [17]. (Here, a maximal trace is either a finite, completed trace or an infinite trace.) The work in [6, 4, 5] models systems with branching as coalgebras of type $\mathsf{T} \circ F$, with $F : \mathsf{Set} \to \mathsf{Set}$ an endofunctor

* S.S. and I.H. are supported by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST, and Grants-in-Aid No. 15KT0012 & 15K11984, JSPS. Part of this work was carried out during a visit of the second author to the University of Southampton, supported by an EPSRC Institutional Sponsorship Award EP/P511407/1.

used to specify linear behaviour (the structure of individual transitions) and $\mathsf{T} : \mathsf{Set} \to \mathsf{Set}$ a monad used to specify branching structure (which typically associates quantities to individual transitions). Examples include non-deterministic/probabilistic/weighted labelled transition systems, with or without explicit termination, but also models with a more general "linear" behaviour, including tree-like behaviour; for example, taking $F : \mathsf{Set} \to \mathsf{Set}$ to be $A \times \mathsf{Id} \times \mathsf{Id}$ with $\mathsf{Id} : \mathsf{Set} \to \mathsf{Set}$ the identity functor captures linear behaviours given by infinite labelled binary trees. In this setting, a canonical definition of linear time behaviour of states in a coalgebra with branching associates, to each state and each possible maximal trace (element of the final F-coalgebra), a quantity measuring the extent of the given state exhibiting that trace, accumulated across all branches [6]. This recovers known concepts of infinite trace semantics, including the possibility (for the finite powerset monad), likelihood (for the sub-probability distribution monad) or minimal cost (for a weighted monad with weights modelling costs) of exhibiting a particular trace.

Coalgebraic linear time logics, interpreted over the same types of models, were studied in [4, 5], with formulas specifying properties of linear behaviour, and with their semantics measuring the extent with which such properties hold in states of coalgebras with branching. The monad T uniformly determines both the domain of truth values for the logics and the choice of propositional operators (e.g. finite disjunctions for non-deterministic systems, sub-convex combinations for probabilistic systems and linear combinations for weighted systems). The modal operators employed arise from the functor F, but their interpretation is based both on a choice of associated predicate liftings for F and on a canonical predicate lifting for T, with the latter being used to accumulate the quantities associated to different branches. As with the notion of linear time behaviour, exactly how this accumulation works depends on the type of branching. For non-deterministic systems, one recovers the *aconjunctive* linear-time μ-calculus. For probabilistic systems, the resulting logics differ from probabilistic variants of LTL or the linear time μ-calculus in their absence of boolean operators. This difference is beneficial, as it results in an expensive automata determinisation step required for model checking such probabilistic variants (see [3, Section 10.3]) being avoided with our logics, without a real loss in expressiveness (see Remark 6). For weighted systems, the logics are similar to previously proposed ones [15] in their absence of conjunction operators. Our choice of propositional operators is further supported by results in [5] on the equivalence of the original, step-wise semantics of the logics with an alternative path-based semantics akin to that of LTL, and by the close connection to quantitative automata described in this paper.

We now turn to the contributions of this paper. A definition of the extent with which *two* states in coalgebras with branching exhibit *similar* linear behaviour was given in [6]. Here we take this further by providing alternative characterisations of notions of *maximal* and *finite trace similarity* between two branching coalgebras. These instantiate to the existence/likelihood/minimal joint cost of a *common* trace, in the case of non-deterministic/probabilistic/weighted branching. When one of the coalgebras models the system of interest and the other captures (un)desirable behaviours, these notions provide the right concepts for automata-theoretic model checking. Our alternative characterisations involve a product construction, and a novel notion of *extent* of a coalgebra with branching.

We then extend these ideas to provide an automata-theoretic characterisation of the semantics of the logics in [4, 5]. For this, we use a generalisation of standard parity automata over words/trees, inspired by recent work on coalgebraic trace semantics for Büchi and parity automata [17]. Our automata are parameterised by (i) a *partial semiring* monad, specifying a type of branching, and (ii) a polynomial endofunctor F, specifying a type of linear behaviour. Key to our approach is the notion of *extent* of a parity automaton, which

provides a *quantitative* notion of acceptance, not of a single maximal trace, but across *all* maximal traces that conform to the automaton. This instantiates to the existence/likelihood/minimal cost of an accepting trace (maximal trace satisfying the parity condition), for non-deterministic/probabilistic/weighted branching (with weights modelling costs), respectively (see Example 17). A key advantage of the *extent-based* characterisation of the semantics is a *localised view* of the satisfaction relation of the logics – fixpoints are computed locally on the reachable part of the product between model and formula automata, rather than globally as predicates on the model state space, as stipulated in the original semantics. Our *generic* translation from formulas to automata resembles known translations for non-deterministic systems (see e.g. [19]), while additionally exploiting the choices made in the semantics of our logics to simplify the construction. A translation from automata to formulas is also sketched.

For non-deterministic branching, Büchi automata over words are as expressive as parity ones [11], while over trees this fails to hold [16]. Here we generalise this result to a quantitative setting, under the additional assumption that the branching semiring is *total*. Given our translations from logics to automata and back, this also yields an important result about our logics, namely that for *word-like* linear behaviour (i.e. F isomorphic to a coproduct of constant and identity functors), the alternation degree 1 fragment of the logics is fully expressive. We stress that our translation from parity to Büchi automata preserves the *quantitative* language. To our knowledge, the only related result in a quantitative setting is a translation from probabilistic Rabin to probabilistic Büchi automata [2], which only preserves the *qualitative* language. Unlike the standard translation from parity to Büchi via Rabin automata [11], which uses the idempotence of disjunction, our translation does *not* require idempotence of the semiring addition. For *partial* semirings which arise as sub-semirings of total semirings satisfying our assumptions (as is the case for probabilistic branching), a similar translation is obtained by moving to the larger semiring; this time, the resulting Büchi automaton has branching as specified by the larger semiring, but can nonetheless be used in the same way, given the preservation of the quantitative language (see Remark 33).

Our final contribution is a naïve algorithm for computing extents of parity automata. This requires an additional assumption on the underlying semiring, satisfied by both non-deterministic branching and a bounded variant of weighted branching, to ensure that the computation of individual fixpoints terminates. To summarise, our contributions are:

1. automata-theoretic characterisations of *finite* and *maximal trace similarity* between states of coalgebras with branching (Theorem 14),
2. a notion of *extent* of a parity automaton, parameterised by a choice of branching monad and linear time behaviour (Definition 16),
3. translations from linear time formulas to automata and back, providing an automata-theoretic characterisation of the quantitative semantics of the logics in [4, 5] (Theorem 24),
4. a translation from quantitative parity to quantitative Büchi automata over words (Theorem 32), rendering the alternation degree 1 fragment of the logics fully expressive,
5. an algorithm for computing extents, similar in complexity to known algorithms for emptiness checking in the non-deterministic case [14].

We assume familiarity with the coalgebraic approach to modelling systems. Section 2 recalls the results of [6, 4, 5], while each subsequent section contains one of the above contributions.

2 Preliminaries

2.1 From Partial Commutative Semirings to Commutative Monads

▶ **Definition 1.** A *partial commutative semiring* is a tuple $S := (S, +, 0, \bullet, 1)$ with $(S, +, 0)$ a partial commutative monoid and $(S, \bullet, 1)$ a commutative monoid, with $\bullet$ distributing over $+$; that is, for all $s, t, u \in S$, $s \bullet 0 = 0$, and whenever $t + u$ is defined, then so is $s \bullet t + s \bullet u$ and moreover $s \bullet (t + u) = s \bullet t + s \bullet u$.

The addition operation of any partial commutative semiring induces a pre-order relation $\sqsubseteq$ on S, given by $x \sqsubseteq y$ iff there exists $z \in S$ with $x + z = y$, and having 0 as its least element.

▶ **Example 2.** Here we consider the *boolean semiring* $(\{0,1\}, \vee, 0, \wedge, 1)$, the (partial) *probabilistic semiring* $([0,1], +, 0, *, 1)$, the *tropical semiring* $\mathbb{N} = (\mathbb{N}^\infty, \min, \infty, +, 0)$ and its bounded variants $S_B = ([0, B]^\infty, \min, \infty, +_B, 0)$ with $B \in \mathbb{N}$, where $m +_B n = \begin{cases} m + n, & \text{if } m + n \leq B \\ \infty, & \text{otherwise} \end{cases}$. The associated pre-orders are $\leq$ on $\{0,1\}$ and $[0,1]$, and $\geq$ on $\mathbb{N}^\infty$ and $[0, B]^\infty$.

We further assume that $\sqsubseteq$ has a top element and is an ω-chain complete as well as ω^{op}-chain complete partial order. This holds in all our examples.

A partial commutative semiring S induces a *semiring monad* $(\mathsf{T}_S, \eta, \mu)$ with

$$\mathsf{T}_S(X) = \{\, \varphi : X \to S \mid \mathsf{supp}(\varphi) \text{ is finite}, \sum_{x \in \mathsf{supp}(\varphi)} \varphi(x) \text{ is defined} \,\}$$

$$\eta_X(x)(y) = \begin{cases} 1 & \text{if } y = x \\ 0 & \text{otherwise} \end{cases}, \qquad \mu_X(\Phi)(x) = \sum_{\varphi \in \mathsf{supp}(\Phi)} \Phi(\varphi) \bullet \varphi(x)$$

where $\mathsf{supp}(\varphi) = \{x \in X \mid \varphi(x) \neq 0\}$ is the *support* of φ. Moreover, the monad T_S above is *strong* and *commutative*, with *strength* $\mathsf{st}_{X,Y} : X \times \mathsf{T}_S Y \to \mathsf{T}_S(X \times Y)$ and *double strength* $\mathsf{dst}_{X,Y} : \mathsf{T}_S X \times \mathsf{T}_S Y \to \mathsf{T}_S(X \times Y)$ given by

$$\mathsf{st}_{X,Y}(x, \psi)(z, y) = \begin{cases} \psi(y) & \text{if } z = x \\ 0 & \text{otherwise} \end{cases}, \qquad \mathsf{dst}_{X,Y}(\varphi, \psi)(z, y) = \varphi(z) \bullet \psi(y)$$

We note that $\mathsf{T}_S 1 = S$, with 1 a final object in Set, and therefore S carries a T_S-algebra structure given by $\mu_1 : \mathsf{T}_S^2 1 \to \mathsf{T}_S 1$. The relationship between monads and partial semirings was studied in [8, 6]. We use semiring monads to model branching, with the semirings in Example 2 modelling non-deterministic, probabilistic and weighted branching. In the latter case, we think of the weights as costs associated to individual system steps.

2.2 Finite and Maximal Traces

A coalgebraic approach to defining maximal traces of coalgebras of type $\mathsf{T}_S \circ F$, with F a *polynomial*[1] endofunctor and $\mathsf{T}_S : \mathsf{Set} \to \mathsf{Set}$ a semiring monad, is described in [6]. The monad T_S is used to specify branching structure, whereas the functor F is used to specify linear behaviour, with the elements of the final F-coalgebra defining individual traces. Since any polynomial endofunctor on Set can be written as a coproduct of finite products of identity

[1] An endofunctor $F : \mathsf{Set} \to \mathsf{Set}$ is *polynomial* if it is constructed from constant and identity functors using finite products and arbitrary coproducts.

functors, we readily assume this shape: $FX = \coprod_{\lambda\in\Lambda} X^{\mathsf{ar}(\lambda)}$ with Λ a set of operators with finite arities. The elements of the final F-coalgebra (initial F-algebra) are potentially infinite (resp. finite) trees with nodes labelled by some λ and having as many outgoing edges as $\mathsf{ar}(\lambda)$. The definition of maximal traces resembles the alternative partition-refinement definition of bisimilarity, but differs from it in two key ways: (i) what is defined is a *trace relation* between states of a $\mathsf{T}_S \circ F$-coalgebra and elements of the final F-coalgebra, and (ii) trace relations are *S-valued relations* that measure, for each state in a coalgebra with branching and each linear behaviour, the *extent* (e.g. ability, likelihood or minimal cost) of that state exhibiting the given linear behaviour. Concretely, for a $\mathsf{T}_S \circ F$-coalgebra $\gamma : C \to \mathsf{T}_S FC$, its maximal traces are given by the greatest fixpoint of the following operator on S-valued relations between C and the carrier of the final F-coalgebra (Z, ζ):

$$\mathsf{Rel}_{C,Z} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FZ} \xrightarrow{L_S} \mathsf{Rel}_{\mathsf{T}_S FC,FZ} \xrightarrow{(\gamma\times\zeta)^*} \mathsf{Rel}_{C,Z} \tag{1}$$

Here, $\mathsf{Rel}_{X,Y}$ is the category of S-valued relations on $X \times Y$, and $\mathsf{Rel}(F) : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{FX,FY}$ "lifts" S-valued relations on $X \times Y$ to S-valued relations on $FX \times FY$, with the help of the semiring multiplication: for $R : X \times Y \to S$, $\mathsf{Rel}(F)(R)$ maps $(\iota_\lambda(x_1, \ldots, x_{\mathsf{ar}(\lambda)}), \iota_{\lambda'}(y_1, \ldots, y_{\mathsf{ar}(\lambda')}))$ to 0 if $\lambda \neq \lambda'$, and $(\iota_\lambda(x_1, \ldots, x_{\mathsf{ar}(\lambda)}), \iota_\lambda(y_1, \ldots, y_{\mathsf{ar}(\lambda)}))$ to $R(x_1, y_1) \bullet \ldots \bullet R(x_{\mathsf{ar}(\lambda)}, y_{\mathsf{ar}(\lambda)})$. Also, $L_S : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{\mathsf{T}_S X,Y}$, called *extension lifting*, takes a relation $R : X \times Y \to S$ to the relation $\mu_1 \circ \mathsf{T}_S R \circ \mathsf{st}'_{X,Y} : \mathsf{T}_S X \times Y \to S$, with $\mathsf{st}'_{X,Y} : \mathsf{T}_S X \times Y \to \mathsf{T}_S(X \times Y)$ the *swapped strength map* of T_S. This choice for L_S is canonical, in the sense that $L_S(R)(\_, y)$ is the unique extension of $R(\_, y)$ to a T_S-algebra homomorphism, for $y \in Y$ (see [6] for details). Concretely, $L_S(R)(\sum_i c_i x_i, y) = \mu_1(\sum_i c_i R(x_i, y))$ for $x_i \in X$ and $y \in Y$. The effect of using L_S above is that the quantity ultimately associated to each pair $(c, z) \in C \times Z$ is accumulated across all branches from c, in a step-wise fashion.

A similar treatment of *finite* traces is obtained by replacing the *final* F-coalgebra (Z, ζ) with the *initial* F-algebra (I, ι), with $\iota : FI \xrightarrow{\simeq} I$, and taking the *least* fixpoint of the following operator on S-valued relations:

$$\mathsf{Rel}_{C,I} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FI} \xrightarrow{L_S} \mathsf{Rel}_{\mathsf{T}_S FC,FI} \xrightarrow{(\gamma\times\iota^{-1})^*} \mathsf{Rel}_{C,I}$$

▶ **Example 3.** When $S = (\{0, 1\}, \vee, 0, \wedge, 1)$ (and thus T_S is isomorphic to the finite powerset monad), the (greatest, resp. least) fixpoints of the previous operators relate a state in a non-deterministic coalgebra with a (maximal, resp. finite) trace iff that state can exhibit the given trace. When $S = ([0, 1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, the (greatest, resp. least) fixpoints give, for each state and each (maximal, resp. finite) trace, the likelihood, resp. minimal cost of that state exhibiting the given trace. The precise shape of a trace is determined by the choice of F; taking $F = 1 + A \times \mathsf{Id}$ captures words over A, whereas $F = 1 + A \times \mathsf{Id} \times \mathsf{Id}$ captures binary trees with non-leaf nodes labelled by A.

2.3 Quantitative Linear Time Logics for Coalgebras

We now recall (a variant of) the logics studied in [4, 5]. They are interpreted over $\mathsf{T}_S \circ F$-coalgebras, with T_S and F as before, and have syntax given by

$$\mu\mathcal{L}^{\mathcal{V}}_{\Lambda} \ni \varphi ::= x \mid [\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)}) \mid \textstyle\sum_{i\in I} c_i \bullet \varphi_i \mid \mu x.\varphi \mid \nu x.\varphi$$

with $x \in \mathcal{V}$, $\lambda \in \Lambda$ and $c_i \in S$ such that $\sum_{i\in I} c_i$ is defined. Here, $\mathcal{V}$ is a set of variables and I is a finite set. Writing $\iota_\lambda : X^{\mathsf{ar}(\lambda)} \to FX$ with $\lambda \in \Lambda$ for the coproduct injections, we define,

for each modal operator $\lambda \in \Lambda$, an *S-valued predicate lifting* $[\![\lambda]\!] : S^{-} \times \ldots \times S^{-} \Rightarrow S^{F-}$ by:

$$[\![\lambda]\!](p_1, \ldots, p_{\mathsf{ar}(\lambda)})(\iota_{\lambda'}(x_1, \ldots, x_{\mathsf{ar}(\lambda')})) = \begin{cases} p_1(x_1) \bullet \ldots \bullet p_{\mathsf{ar}(\lambda)}(x_{\mathsf{ar}(\lambda)}), \text{ if } \lambda = \lambda' \\ 0, \text{ otherwise} \end{cases} \tag{2}$$

▶ **Definition 4.** For a $\mathsf{T}_S \circ F$-coalgebra (C, γ) and a valuation $V : \mathcal{V} \to S^C$, the denotation $[\![\varphi]\!]^V_\gamma \in S^C$ of a formula $\varphi \in \mu\mathcal{L}^{\mathcal{V}}_\Lambda$ is defined inductively on the structure of φ by

- $[\![x]\!]^V_\gamma = V(x)$,
- $[\![\sum_{i\in I} c_i \bullet \varphi_i]\!]^V_\gamma = \mu_1(\sum_{i\in I} c_i\, [\![\varphi_i]\!]^V_\gamma)$,
- $[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]^V_\gamma = \gamma^*(\mathsf{ext}_{FC}([\![\lambda]\!]_C([\![\varphi_1]\!]^V_\gamma, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]^V_\gamma)))$, where the *extension predicate lifting* $\mathsf{ext} : S^{-} \Rightarrow S^{\mathsf{T}_S -}$ takes an S-valued predicate $p : C \to S$ to the S-valued predicate $\mu_1 \circ \mathsf{T}_S p : \mathsf{T}_S C \to S$, while $\gamma^* : S^{\mathsf{T}_S FC} \to S^C$ is pre-composition with γ.
- $[\![\mu x.\varphi]\!]^{V\setminus\{x\}}_\gamma$ ($[\![\nu x.\varphi]\!]^{V\setminus\{x\}}_\gamma$) is the least (resp. greatest) fixpoint of the operator on S^C taking $p : C \to S$ to $[\![\varphi]\!]^{V[p/x]}_\gamma$, where the valuation $V[p/x] : \mathcal{V} \to S^C$ takes x to p and $y \in \mathcal{V} \setminus \{x\}$ to $V(y)$.

(In the second clause, both the formal sum notation and the action of μ_1 have been extended pointwisely to functions on C.) We write $\mu\mathcal{L}_\Lambda$ for the set of *closed* formulas ($\mathcal{V} = \emptyset$).

For the operator in the last clause of Definition 4 to be order-preserving, monotonicity of both ext and $[\![\lambda]\!]$, with $\lambda \in \Lambda$, is required, and proved in [5]. The existence of lfp*s*, respectively gfp*s* then follows by [10, Theorem 8.22], which assumes an order-preserving operator on a cpo. The use of extension lifting in the third clause of Definition 4 results in $[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]^V_\gamma(c)$ accumulating the values $[\![\lambda]\!]_C([\![\varphi_1]\!]^V_\gamma, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]^V_\gamma)(f_i)$, with $\gamma(c) = \sum_i c_i f_i$, by taking into account the weights c_i:

$$[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]^V_\gamma(c) = \mu_1(\sum_i c_i [\![\lambda]\!]_C([\![\varphi_1]\!]^V_\gamma, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]^V_\gamma)(f_i))$$

The presence of weighted sums in the logics is supported by results in [5] showing that the inclusion of such sums preserves the equivalence of the above step-wise semantics with an alternative, *path-based* semantics, akin to that of LTL. Finite conjunctions are missing from the logics, and our step-wise semantics prevents their inclusion. It is worth noting, however, that fixpoint logics for weighted systems are similar in their absence of conjunctions [15]. For *total* semirings S, finite disjunctions are present as finite weighted sums with the weights equal to 1. Their interpretation is as expected: the formula $\sum_i 1 \bullet \varphi_i$ (written more simply $\sum_i \varphi_i$) measures the extent of conforming to one of the φ_is.

▶ **Example 5.** Taking $F = 1 + A \times \mathsf{Id} \simeq 1 + \coprod_{a\in A} \mathsf{Id}$ yields a logic with a nullary modality $*$ (for termination), and unary modalities $[a]$ with $a \in A$, with the usual interpretation. Taking $F = A \times \mathsf{Id} \times \mathsf{Id} \simeq \coprod_{a\in A}(\mathsf{Id} \times \mathsf{Id})$ yields a logic with binary modalities $[a]$ with $a \in A$, with associated predicate liftings given by $[\![a]\!]_X(p_1, p_2)(\iota_{a'}(x, y)) = \begin{cases} p_1(x) \bullet p_2(y), \text{ if } a' = a \\ 0, \text{ otherwise} \end{cases}$, for $p_1, p_2 \in S^X$ and $x, y \in X$. Irrespective of the choice of F, when $S = (\{0, 1\}, \vee, 0, \wedge, 1)$, a formula φ holds in a state of a $\mathsf{T}_S \circ F$-coalgebra iff that state admits a maximal trace satisfying φ. For $S = ([0, 1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, $[\![\varphi]\!]_\gamma : C \to S$ measures the likelihood, resp. minimal cost of states of $\mathsf{T}_S \circ F$-coalgebras conforming to φ (suitably scaled according to the weighted sums present in φ).

▶ Remark 6. Taking $S = ([0, 1], +, 0, *, 1)$ and $F = 1 + A \times \mathsf{Id}$ yields a linear time logic for probabilistic transition systems. The absence of pure disjunctions makes this logic different

from probabilistic LTL. A variant of our logics which incorporates disjunctions that can be resolved in one step (e.g. $[a]\varphi \vee [b]\psi$ with $a \neq b$) as new modalities turns out to be more expressive than probabilistic LTL (see [5, Example 4.3]). In this case, deterministic parity automata (known to be more expressive than LTL) can be directly encoded in the logic.

▶ Remark 7. By casting our logics into a dual adjunction framework, as done in [5], it follows immediately that $\mathsf{T}_S \circ F$-behavioural equivalence implies logical equivalence. This, however, is not very interesting, given the linear time nature of our logics. A detailed study of a weaker, trace-based notion of equivalence for which our logics are both sound and expressive is left as future work.

2.4 Equational Systems

The use of nested fixpoints in the semantics of fixpoint logics can elegantly be rephrased in terms of solutions of equational systems [1].

▶ **Definition 8.** An *equational system* over posets $L_1, \ldots, L_n$ is a sequence of equations $u_1 =_{\eta_1} f_1(u_1, \ldots, u_n)$, $\ldots$, $u_n =_{\eta_n} f_n(u_1, \ldots, u_n)$, where $u_1, \ldots, u_n$ are *variables*, $\eta_i \in \{\mu, \nu\}$ and $f_i : L_1 \times \ldots \times L_n \to L_i$ is a monotone function. A variable u_j is a *μ-variable* (*ν-variable*) if $\eta_j = \mu$ (resp. $\eta_j = \nu$).

A precise definition of the solution of an equational system can be found in [1, Section 1.4.4]. Intuitively, assuming that the L_is have enough suprema and infima, the *solution* of an equational system is defined as follows:

1. the first equation is solved (by taking either the least or the greatest solution, depending on η_1), to obtain an interim solution $u_1 = l_1^{(1)}(u_2, \ldots, u_n)$;
2. this is substituted for u_1 in the second equation, yielding a new equation $u_2 =_{\eta_2} f_2^{\sharp}(u_2, \ldots, u_n)$;
3. the second equation is solved to obtain an interim solution $u_2 = l_2^{(2)}(u_3, \ldots, u_n)$;
4. continuing this way from left to right eventually eliminates all the variables and leads to a closed solution $u_n = l_n^{(n)} \in L_n$; and
5. closed solutions are propagated back from right to left to yield closed solutions for all of $u_1, \ldots, u_n$.

Instead of μ- and ν-annotations, some of the equational systems appearing later in the paper use natural numbers, with odd (even) values indicating μ- (resp. ν-) variables, and with the order of equations being determined by the natural order on $\mathbb{N}$. We also use a *generalised form* for equational systems, which allows several equations indexed by the same value, all of which are to be solved simultaneously.

3 An Automata-Based Approach to Trace Similarity

As already sketched in [6], notions of *maximal* and resp. *finite trace similarity* between states of $\mathsf{T}_S \circ F$-coalgebras can be defined by using a *double extension lifting* in place of the extension lifting L_S of (1). This section paves the way towards an automata-based characterisation of the semantics of the logic $\mu\mathcal{L}_\Lambda$, by providing a similar (and simpler) characterisation of maximal and resp. finite trace similarity.

The *double extension lifting* $L'_S : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{\mathsf{T}_S X, \mathsf{T}_S Y}$ takes an S-valued relation $R : X \times Y \to S$ to the S-valued relation $\mu_1 \circ \mathsf{T}_S R \circ \mathsf{dst}_{X,Y} : \mathsf{T}_S X \times \mathsf{T}_S Y \to S$. Compared to L_S, L'_S uses the double strength map of T_S in place of the swapped strength map to yield

a relation on $\mathsf{T}_S X \times \mathsf{T}_S Y$. As with L_S, this choice for L'_S is canonical (see [6]), and satisfies $L'_S(R)(\sum_i c_i x_i, \sum_j d_j y_j) = \mu_1(\sum_i \sum_j (c_i \bullet d_j) R(x_i, y_j))$.

▶ **Definition 9.** The *maximal* (resp. *finite*) *trace similarity relation* between two $\mathsf{T}_S \circ F$-coalgebras (C, γ) and (D, δ) is the greatest (resp. least) fixpoint of the following operator on S-valued relations between C and D:

$$\mathsf{Rel}_{C,D} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FD} \xrightarrow{L'_S} \mathsf{Rel}_{\mathsf{T}_S FC, \mathsf{T}_S FD} \xrightarrow{(\gamma \times \zeta)^*} \mathsf{Rel}_{C,D}$$

We write $\simeq^{\nu}_{\gamma,\delta} \colon C \times D \to S$ and $\simeq^{\mu}_{\gamma,\delta} \colon C \times D \to S$ for these relations.

▶ **Example 10.** For $S = (\{0,1\}, \vee, 0, \wedge, 1)$, maximal (finite) trace similarity relates precisely those states which admit a common maximal (resp. finite) trace. For $S = ([0,1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, maximal (finite) trace similarity measures the likelihood, resp. minimal joint cost of two states exhibiting a common maximal (resp. finite) trace.

We now introduce notions of ν- and μ-*extent* of a $\mathsf{T}_S \circ F$-coalgebra, and rephrase the definitions of $\simeq^{\nu}_{\gamma,\delta}$ and $\simeq^{\mu}_{\gamma,\delta}$ using these notions. The idea is to measure the weight with which a state in a coalgebra with branching can exhibit *any* maximal (resp. finite) trace. This weight is cumulative across all the branches.

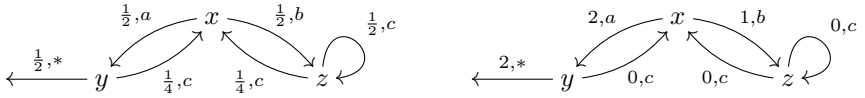
▶ **Definition 11.** The ν-*extent* (μ-*extent*) of a $\mathsf{T}_S \circ F$-coalgebra (A, α) is the gfp (resp. lfp) of the operator on S^A taking $p : A \to S$ to the composition

$$A \xrightarrow{\alpha} \mathsf{T}_S FA \xrightarrow{\mathsf{T}_S Fp} \mathsf{T}_S FS \xrightarrow{\mathsf{T}_S(\bullet_F)} \mathsf{T}_S S = \mathsf{T}_S^2 1 \xrightarrow{\mu_1} \mathsf{T}_S 1 = S$$

where $\bullet_F : FS \to S$ is given by $\bullet_F(\iota_\lambda(s_1, \ldots, s_{\mathsf{ar}(\lambda)})) = s_1 \bullet \ldots \bullet s_{\mathsf{ar}(\lambda)}$ for $\lambda \in \lambda$.

The above operator uses a one-step unfolding of the coalgebra structure to compute a finer approximation of the extent on a state based on the extent on its immediate successors. As the generality of F allows for immediate successors which are tuples of states, the semiring multiplication may also need to be used (in $\bullet_F$). The monad multiplication is used to accumulate the values from different branches. The composition in Definition 11 takes $a \in A$ with $\alpha(a) = \sum_i c_i(a_i^1, \ldots, a_i^{j_i})$ to $\mu_1(\sum_i c_i(p(a_i^1) \bullet \ldots \bullet p(a_i^{j_i})))$. That is, the extent associated to a particular state accumulates the extents associated to its immediate successors, scaled by the weights of the corresponding branches.

▶ **Example 12.** For the $\mathsf{T}_S \circ F$-coalgebras below, with $F = 1 + A \times \mathsf{Id}$ and $S = ([0,1], +, 0, *, 1)$ (resp. $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$), the ν-extent maps x to 0.4, y to 0.6 and z to 0.2 (resp. x and y to 1 and z to 0), whereas the μ-extent again maps x to 0.4, y to 0.6 and z to 0.2 (resp. x and z to 4 and y to 2). Intuitively, the reason for the μ- and ν-extents being the same in the probabilistic case is that the likelihood of never reaching y from either x or z is 0.



The notions of ν- and μ-extent turn out to be particularly useful when applied to the *product* of two $\mathsf{T}_S \circ F$-coalgebras; this collects their common F-behaviour, suitably quantified with the help of the monad structure of T_S.

▶ **Definition 13.** The *product* of $\mathsf{T}_S \circ F$-coalgebras (C, γ) and (D, δ) is the $\mathsf{T}_S \circ F$-coalgebra with carrier $C \times D$ and transition function $\gamma \otimes \delta$ given by

$$C \times D \xrightarrow{\gamma \times \delta} \mathsf{T}_S FC \times \mathsf{T}_S FD \xrightarrow{\mathsf{dst}_{FC,FD}} \mathsf{T}_S(FC \times FD) \xrightarrow{\langle F\pi_1, F\pi_2\rangle^*} \mathsf{T}_S F(C \times D)$$

where $\langle F\pi_1, F\pi_2\rangle^*$ is pre-composition with $\langle F\pi_1, F\pi_2\rangle : F(C \times D) \to FC \times FD$.

The effect of pre-composing with $\langle F\pi_1, F\pi_2\rangle$ is that pairs of non-matching one-step behaviours are discarded from the resulting coalgebra.

Our first result relates the ν- and μ-extents of the product of two coalgebras with the maximal and respectively finite trace similarity relation between them.

▶ **Theorem 14.** *Let (C, γ) and (D, δ) be two $\mathsf{T}_S \circ F$-coalgebra. The ν-extent (μ-extent) of the product coalgebra $(C \times D, \gamma \otimes \delta)$ coincides with the maximal trace similarity relation $\simeq^{\nu}_{\gamma,\delta}$ (resp. the finite trace similarity relation $\simeq^{\mu}_{\gamma,\delta}$).*

Proof (sketch). The proof involves showing that the operators used in the definition of $\simeq^{\nu}_{\gamma,\delta}$ and $\simeq^{\mu}_{\gamma,\delta}$ on the one hand, and of the ν-/μ-extent of the product automaton on the other, coincide. ◀

4 Parity (S, F)-Automata and their Extent

We now consider *parity* (S, F)*-automata*, as extensions of $\mathsf{T}_S \circ F$-coalgebras with a parity map, and define their *extent*. Parity maps are a natural way to formulate acceptance conditions for automata on infinite words/trees: the states of the automaton are assigned natural number *parities*, and a run of the automaton is *accepting* iff the largest parity occurring infinitely often along the run is even (see e.g. [11]). We use parity maps in a similar way, but this time in a quantitative setting where the emphasis moves away from individual runs.

▶ **Definition 15.** A *parity* (S, F)*-automaton* is given by a $T_S \circ F$-coalgebra (A, α) together with a function $\Omega : A \to \{1, 2, \ldots\}$ with finite range, called *parity map*.

A similar notion called (T, F)*-system* was considered in [17], albeit under different assumptions on the monad T, which rule out semiring monads as considered here. A trace semantics for a (T, F)-system was defined in loc. cit. as a Kleisli map obtained by taking least and greatest fixpoints, and this was proved to instantiate to the standard notions of acceptance by a non-deterministic, resp. probabilistic parity automaton. As in [17], our assignment of parities to *all* states of the automaton allows for a smooth relationship to equational systems.

The *extent* of a parity automaton generalises the ν- and μ-extents of a coalgebra by taking into account the different parities associated to the automaton states. The only difference is that now the extent involves a collection of *nested* fixpoints, one for each parity.

▶ **Definition 16.** Let (A, α, Ω) be a parity (S, F)-automaton with $\mathsf{ran}(\Omega) \subseteq \{1, \ldots, n\}$, let $A_k = \{a \in A \mid \Omega(a) = k\}$, and let $\alpha_k = \alpha \circ \iota_k : A_k \to \mathsf{T}_S FA$ denote the restriction of α to A_k. The *extent* $e = [e_1, \ldots, e_n] : A \to S$ *of* (A, α, Ω) is the solution of the equational system

$$\begin{bmatrix} u_1 & =_\mu & \mu_1 \circ \mathsf{T}_S(\bullet_F) \circ T_S F[u_1, \ldots, u_n] \circ \alpha_1 \\ & \vdots & \\ u_n & =_\eta & \mu_1 \circ \mathsf{T}_S(\bullet_F) \circ T_S F[u_1, \ldots, u_n] \circ \alpha_n \end{bmatrix} \tag{3}$$

with $\eta = \mu$ ($\eta = \nu$) if n is odd (resp. even), with variables u_k ranging over the poset $(S^{A_k}, \sqsubseteq)$ (and therefore $[u_1, \ldots, u_n] : A \to S$), and with the rhss of the equations pictured below:

$$A_k \xrightarrow{\alpha_k} \mathsf{T}_S FA \xrightarrow{\mathsf{T}_S F[u_1,\ldots,u_n]} \mathsf{T}_S FS \xrightarrow{\mathsf{T}_S(\bullet_F)} \mathsf{T}_S S = \mathsf{T}_S^2 1 \xrightarrow{\mu_1} \mathsf{T}_S 1 = S$$

▶ **Example 17.** For non-deterministic/probabilistic/weighted systems, the extent captures the existence/likelihood/minimal cost of an accepting run. For the coalgebras in Example 12, assigning parities 1 to the states x and y and 2 to the state z yields automata with extents mapping x to 0.4, y to 0.6 and z to 0.2, and resp. x to 1, y to 1 and z to 0. The reason for the extent being similar to the μ- and ν-extents of the underlying coalgebra in the probabilistic case is that, although runs which visit z infinitely often contribute to the extent, under the current assignment of probabilities to transitions, the contributed quantity is 0. The situation would be different if the transition from z to x was removed and the one from z to itself had probability 1, in which case the extent would map x to $\frac{12}{14}$, y to $\frac{5}{7}$ and z to 1.

▶ Remark 18. A trace semantics for parity (S, F)-automata similar to that of [17] can also be defined, using an equational system whose variables u_k range over $S^{A_k \times Z}$. In this case, the rhs of the kth equation is given by:

$$\begin{array}{ccccccc} A_k \times Z & \xrightarrow{\alpha\times(\eta_{FZ}\circ\zeta)} & \mathsf{T}_S FA \times \mathsf{T}_S FZ & \xrightarrow{\mathsf{dst}} & \mathsf{T}_S(FA \times FZ) & \xrightarrow{\langle F\pi_1, F\pi_2\rangle^*} & \mathsf{T}_S F(A \times Z) \\ & & & & & & \downarrow{\scriptstyle \mathsf{T}_S F[u_1,\ldots,u_n]} \\ S & \xleftarrow[\mu_1]{} & \mathsf{T}_S S & \xleftarrow[\mathsf{T}_S(\bullet_F)]{} & \mathsf{T}_S FS & & \end{array}$$

5 From Linear Time Logics to Parity (S, F)-Automata

We now show how to assign, to each *clean* and *guarded* formula of $\mu\mathcal{L}_\Lambda$ (Definition 19), a parity (S, F)-automaton. We then give an automata-theoretic characterisation of the semantics of a $\mu\mathcal{L}_\Lambda$-formula, using the extent of a product automaton (between a model and a formula automaton). The next definition is standard for fixpoint logics (see e.g. [18]).

▶ **Definition 19.** For a set $\mathcal{V}$ of variables, a formula $\varphi \in \mu\mathcal{L}_\Lambda^{\mathcal{V}}$ is *clean* if no variable appears both free and bound, or is bound more than once, in φ, and *guarded* if each occurrence of a bound variable inside its defining fixpoint formula lies within the scope of a modal operator.

For a clean formula $\varphi \in \mu\mathcal{L}_\Lambda^{\mathcal{V}}$, we write $\mathsf{BVar}(\varphi)$ for its set of bound variables. Also, for $x, y \in \mathsf{BVar}(\varphi)$, we write $\varphi_x = \eta x.\psi_x$ for the unique sub-formula of φ which binds x, and $y \leq x$ iff φ_y is a sub-formula of φ_x. Our technical development will require first transforming a guarded formula to a *strictly guarded* one, as defined below.

▶ **Definition 20.** A formula $\varphi \in \mu\mathcal{L}_\Lambda^{\mathcal{V}}$ is (i) *strongly guarded* if every occurrence of a fixpoint variable x inside the defining formula ψ_y of a variable y (with $y \leq x$) appears within the scope of a modal operator, and (ii) *strictly guarded* if every occurrence of a fixpoint variable x inside ψ_x is immediately preceded by a modal operator.

Guardedness requires that, in the formula syntax tree, one cannot pass from a fixpoint quantification $\eta y.\psi_y$ to an occurrence of y without encountering a modal operator. Strong guardedness additionally requires that this is the case when passing from $\eta y.\psi_y$ to *any* fixpoint variable x occurring inside ψ_y. It is easy to see that every guarded formula is equivalent to a strongly guarded one. To see this, assume for simplicity that $S = (\{0, 1\}, \vee, 0, \wedge, 1)$ and therefore the logic only contains non-weighted sums (i.e. disjunctions) which we denote by $+$. Then, a non-strongly guarded occurrence of variable x, necessarily of the form $\eta x.\varphi[\eta' y.(x+\psi)]$ with φ a formula with a guarded hole, is equivalent to $\eta x.\varphi[x + \eta' y'.\psi[(x + y')/y]]$, where x is now guarded in $\psi[(x + y')/y]$ as y was initially guarded in ψ. (This argument generalises to weighted sums, where now $x+\psi$ is replaced by $\sum_i c_i\psi_i$ with $\psi_i = x$ for some i. However, when

S is a *partial* semiring, an additional unfolding of the formula $\eta' y'.\psi[(x + y')/y]$ is required, as the counterpart of the sum $x + \eta' y'.\psi[(x + y')/y]$ may not be defined.) Strict guardedness additionally requires that occurrences of x inside ψ_y are *immediately* preceded by modal operators. Given the distributivity of modal operators over weighted sums (an immediate consequence of the definition of $[\![\lambda]\!]$), one can translate a strongly guarded formula into an equivalent, strictly guarded one by pushing weighted sums outside the modal operators.

The next definition can be traced back to the notion of *Fischer-Ladner closure* [12].

▶ **Definition 21.** A set $C \subseteq \mu\mathcal{L}_\Lambda^{\mathcal{V}}$ of formulas is *closed* if

- $\psi[\eta x.\psi/x] \in C$ whenever $\eta x.\psi \in C$, for $\eta \in \{\mu, \nu\}$,
- $\varphi_i \in C$ for $i \in \{1, \dots, n\}$, whenever $\sum_{i\in\{1,\dots,n\}} c_i \bullet \varphi_i \in C$,
- $\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)} \in C$ whenever $[\lambda](\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)}) \in C$.

The *closure* $\mathsf{Cl}(\varphi)$ of a $\mu\mathcal{L}_\Lambda^{\mathcal{V}}$-formula φ is the smallest closed set containing φ.

We note that $\mathsf{Cl}(\varphi)$ is always finite: the second and third clauses above can only be applied finitely many times before the first clause applies; and applications of the first clause only lead to the inclusion of a new formula in $\mathsf{Cl}(\varphi)$ once for each fixpoint sub-formula of φ.

We now exploit the close relationship between the logic $\mu\mathcal{L}_\Lambda$ on the one hand and the semiring S and endofunctor F on the other to associate, to each clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$, a parity (S, F)-automaton with carrier $\mathsf{Cl}(\varphi)$. To this end, we assign natural numbers n_x to variables $x \in \mathsf{BVar}(\varphi)$ in a way which is consistent with the order $\mathsf{BVar}(\varphi)$, that is, $n_x \leq n_y$ whenever $x \leq y$, and which differentiates between least and greatest fixpoints, that is, n_x is odd (even) if $\varphi_x = \mu x.\psi$ (resp. $\varphi_x = \nu x.\psi$). The number of parities can be optimised to equal the *alternation depth* of φ, given by the number of alternations between least and greatest fixpoints [1]. Hereafter we assume that φ is a fixpoint formula (otherwise the formula $\nu x.\varphi$ with x a fresh variable can be considered instead).

▶ **Definition 22.** The *parity* (S, F)*-automaton* $(\mathsf{Cl}(\varphi), \beta, \Omega)$ associated to a clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$ with $\varphi = \varphi_z = \eta z.\psi_z$ has $\beta : \mathsf{Cl}(\varphi) \to \mathsf{T}_S F\mathsf{Cl}(\varphi)$ and $\Omega : \mathsf{Cl}(\varphi) \to \{1, 2, \dots\}$ defined by induction on the structure of $\mathsf{Cl}(\varphi)$:

- $\Omega(\eta x.\psi_x) = n_x$ (and hence $\Omega(\varphi) = n_z$),
- $\beta(\eta x.\psi_x) = \beta(\psi[\eta x.\psi_x/x])$; $\Omega(\psi[\eta x.\psi/x]) = n_x$, unless $\psi[\eta x.\psi/x]$ is a fixpoint formula itself, in which case the previous clause applies,
- $\beta\big(\sum_{i\in\{1,\dots,n\}} c_i \bullet \varphi_i\big) = \mu_{F\mathsf{Cl}(\varphi)}\big(\sum_{i\in\{1,\dots,n\}} c_i\, \beta(\varphi_i)\big)$; $\Omega(\varphi_i) = \Omega(\sum_{i\in\{1,\dots,n\}} c_i \bullet \varphi_i)$ for $i \in \{1, \dots, n\}$, unless φ_i is a fixpoint formula, in which case the first clause applies,
- $\beta([\lambda](\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)})) = \eta_{F\mathsf{Cl}(\varphi)}(\iota_\lambda(\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)}))$; $\Omega(\varphi_i) = \Omega([\lambda](\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)}))$ for $i \in \{1, \dots, \mathsf{ar}(\lambda)\}$, unless φ_i is a fixpoint formula, in which case the first clause applies,

with $\eta : \mathsf{Id} \Rightarrow \mathsf{T}_S$ and $\mu : \mathsf{T}_S^2 \Rightarrow \mathsf{T}_S$ the unit and resp. multiplication of T_S. (Note the difference in denotations between the first and the second occurrences of the $\sum$ symbol in the third clause: the first is part of a propositional symbol of $\mu\mathcal{L}_\Lambda$, whereas the second describes an element of $\mathsf{T}_S\mathsf{T}_S F\mathsf{Cl}(\varphi)$ as a formal sum.)

Thus, formulas in $\mathsf{Cl}(\varphi)$ are assigned parities starting from the outermost formula, and with a formula receiving the same parity as the smallest formula which contains it – unless the given formula is a fixpoint one $\eta_x.\psi_x$, in which case its parity is n_x. This is standard (see e.g. [7]), and gives $\Omega(\psi') \leq \Omega(\psi)$ whenever ψ' is a sub-formula of ψ different from a fixpoint formula. However, if a formula $\psi \in \mathsf{Cl}(\varphi)$ occurs several times within φ, $\Omega(\psi)$ is defined more than once according to the above definition. In this case, a copy of ψ should be made within $\mathsf{Cl}(\varphi)$ for each occurrence of ψ, with different copies assigned possibly different

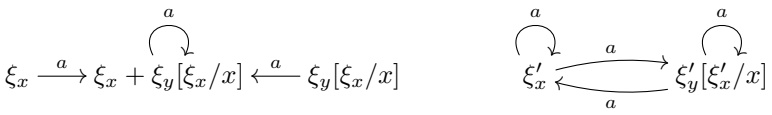
parities. To avoid complicating the definition of the parity automaton associated to φ, and our subsequent exposition, we assume (for the above definition) that φ does not contain several occurrences of any sub-formula. Our technical development does not, however, depend on this assumption.

The recursive definition of β, with base case $[\lambda](\varphi_1, \dots, \varphi_{\mathsf{ar}(\lambda)})$, deviates from the more standard approach to translating fixpoint formulas to automata (see e.g. [18]), which applies more generally to unguarded formulas and involves an intermediate automaton with silent steps. Here, silent steps are automatically absorbed into the next non-silent transition, with no unwanted consequences. This is possible due to our strict guardedness assumption:

1. Guardedness ensures well-definedness of β, as only a finite number of applications of the second and third clauses of Definition 19 are possible before the last clause applies.
2. Strict guardedness ensures that the implicit elimination of silent steps is such that every path from φ_y to x in the formula syntax tree corresponds to a "path" from $\varphi_y[\varphi_x/x]$ to φ_x in the resulting automaton. Thus, states (such as φ_x) with parities greater than the parity associated to φ_y are not sidestepped during the implicit elimination of silent steps. The first formula of Example 23 illustrates why strict guardedness is needed.

▶ **Example 23.** Assume $F = 1 + A \times \mathsf{Id}$. Thus, the associated logic contains a single nullary modality $*$ and unary modalities $[a]$ with $a \in A$.

1. The strongly guarded formula $\xi_x := \nu x.\xi_y$ with $\xi_y := \mu y.[a](x + y)$, with associated automaton shown below on the left, is not strictly guarded. This automaton does not faithfully represent ξ_x, as it does not accept a^ω (since $\Omega(\xi_x + \xi_y[\xi_x/x]) = \Omega(\xi_y) = 1$). On the other hand, the automaton associated to the equivalent, strictly guarded formula $\xi'_x := \nu x.\xi'_y$ with $\xi'_y := \mu y.([a]x + [a]y)$, shown below on the right, correctly accepts a^ω (as $\Omega(\xi'_x) = 2$). The problem with ξ_x is that the path from ξ_y to x in the formula syntax tree does not correspond to a path from $\xi_x + \xi_y[\xi_x/x]$ to ξ_x in the resulting automaton (and similarly for the path from ξ_x to x).



2. The automata associated to the strictly guarded formulas $\varphi_x := \nu x.(\varphi_y + [b]x)$ with $\varphi_y := \mu y.(* + [a]y)$, and $\psi_x := \nu x.\psi_y$ with $\psi_y := \mu y.[c]([a]x + [b]y)$, are:

with $\psi'_y := \psi_y[\psi_x/x]$, $\Omega(\varphi_x) = \Omega(\psi_x) = 2$ and $\Omega(\varphi_y) = \Omega(\psi'_y) = \Omega([a]\psi_x + [b]\psi'_y) = 1$.

Given a $\mathsf{T}_S \circ F$-coalgebra (C, γ) and $\varphi \in \mu\mathcal{L}_\Lambda$ with associated automaton $(\mathsf{Cl}(\varphi), \beta, \Omega)$, one can endow the product of (C, γ) and $(\mathsf{Cl}(\varphi), \beta)$ with a parity map by setting $\Omega(c, \psi) = \Omega(\psi)$ for $(c, \psi) \in C \times \mathsf{Cl}(\varphi)$. The next result provides a characterisation of $[\![\varphi]\!]_\gamma$ using the extent of the resulting parity automaton.

▶ **Theorem 24.** *If $(\mathsf{Cl}(\varphi), \beta, \Omega)$ with $\mathsf{ran}(\Omega) \subseteq \{1, \dots, n\}$ is the parity automaton for a clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$, (C, γ) is a $\mathsf{T}_S \circ F$-coalgebra and $e = [(e_n)_{n \in \mathsf{ran}(\Omega)}] : A \to S$ is the extent of the product parity automaton (A, α, Ω) of (C, γ) and $(\mathsf{Cl}(\varphi), \beta, \Omega)$, then $[\![\varphi]\!]_\gamma(c) = e(c, \varphi)$ for $c \in C$.*

The proof of Theorem 24 involves defining an equational system E_φ in generalised form (see Section 2.4), whose solution is known to provide an alternative characterisation of $[\![\varphi]\!]_\gamma$, and proving that this solution can alternatively be characterised using the extent of the product automaton (A, α, Ω). This intermediary result makes use of solution-preserving substitutions of rhs*s* of equations in E_φ for the respective variables, to transform E_φ into an equational system equivalent to the one used to define the extent of the product automaton.

The equational system E_φ employs a variable u_ψ ranging over S^C for each formula $\psi \in \mathsf{Cl}(\varphi)$. In order to specify the rhs*s* of E_φ, we define, for each $\psi \in \mathsf{Cl}(\varphi)$, a term $d_\psi : (S^C)^{\mathsf{Cl}(\varphi)} \to S^C$ over these variables:

$$\begin{aligned} d_{[\lambda](\varphi_1,\ldots,\varphi_{\mathsf{ar}(\lambda)})} &= \gamma^*(\mathsf{ext}_{FC}([\![\lambda]\!](u_{\varphi_1},\ldots,u_{\varphi_{\mathsf{ar}(\lambda)}}))),\\ d_{\sum_{i\in\{1,\ldots,n\}} c_i \bullet \varphi_i} &= \mu_1(\sum_{i\in\{1,\ldots,n\}} c_i u_{\varphi_i}), \\ d_{\eta x.\psi} &= u_{\eta x.\psi}. \end{aligned} \tag{4}$$

▶ **Definition 25.** For $\varphi \in \mu\mathcal{L}_\Lambda$ clean and strictly guarded, the *system* E_φ collects (i) all equations $u_{\eta x.\psi} =_{\Omega(\eta x.\psi)} d_{\psi[\eta x.\psi/x]}$ with $\eta x.\psi \in \mathsf{Cl}(\varphi)$, and (ii) all equations $u_\xi =_{\Omega(\xi)} d_\xi$ with $\xi \in \mathsf{Cl}(\varphi)$, $\xi \neq \eta x.\psi$, where the u_ψs range over S^C for $\psi \in \mathsf{Cl}(\varphi)$.

The following result is folklore (see e.g. [7] for a similar result).

▶ **Proposition 26.** *For $\varphi \in \mu\mathcal{L}_\Lambda$ clean and guarded, and (C, γ) a $\mathsf{T}_S \circ F$-coalgebra, let $(v_\psi)_{\psi\in\mathsf{Cl}(\varphi)}$ denote the solution of the equational system E_φ. Then, for $\psi \in \mathsf{Cl}(\varphi)$, $[\![\psi]\!]_\gamma : C \to S$ coincides with $d_\psi[(v_\xi/u_\xi)_{\xi\in\mathsf{Cl}(\varphi)}]$.*

▶ **Example 27.** The systems of equations associated to ξ_x and resp. φ_x of Example 23 are

$$\begin{bmatrix} u_{\xi_x} & =_2 & u_{\xi_y[\xi_x/x]} \\ u_{\xi_y[\xi_x/x]} & =_1 & \gamma^*(\mathsf{ext}_{FC}([\![a]\!]u_{\xi_x+\xi_y[\xi_x/x]})) \\ u_{\xi_x+\xi_y[\xi_x/x]} & =_1 & \mu_1(u_{\xi_x} + u_{\xi_y[\xi_x/x]}) \end{bmatrix}, \quad \begin{bmatrix} u_{\varphi_x} & =_2 & \mu_1(u_{\varphi_y} + u_{[b]\varphi_x}) \\ u_{[b]\varphi_x} & =_2 & \gamma^*(\mathsf{ext}_{FC}([\![b]\!]u_{\varphi_x})) \\ u_{\varphi_y} & =_1 & \mu_1(u_* + u_{[a]\varphi_y}) \\ u_* & =_1 & \gamma^*(\mathsf{ext}_{FC}([\![*]\!])) \\ u_{[a]\varphi_y} & =_1 & \gamma^*(\mathsf{ext}_{FC}([\![a]\!]u_{\varphi_y})) \end{bmatrix}$$

The definitions of both E_φ and β are driven by the structure of $\mathsf{Cl}(\varphi)$, and use the same strategy to associate parities to formulas. However, the definition of β sidesteps certain formulas during the implicit elimination of silent steps from the resulting automaton. The proof of Theorem 24, not included here for space reasons, shows that, under the assumption that φ is strictly guarded, carrying out a suitable choice of substitutions (those implicitly performed in the definition of β) on E_φ results in an equational system equivalent to both E_φ and the system of equations defining the extent of the product automaton.

For the modal μ-calculus, a converse translation, from parity automata to fixpoint formulas, also exists. This is defined by induction on the number of parities and uses *vectorial syntax* as an intermediary step. Vectorial syntax [1] generalises standard fixpoint calculus syntax by replacing fixpoint variables with arrays of such variables, all with the same parity, with the corresponding fixpoints being computed simultaneously. A similar translation from parity (S, F)-automata to $\mu\mathcal{L}_\Lambda$-formulas can be defined here. A translation from a parity (S, F)-automaton to vectorial syntax is straightforward: each automaton state yields a new variable, with parity given by the automaton, and whose defining formula is taken from the coalgebra map of the automaton. A translation from vectorial to standard syntax is then carried out by appealing to the Bekič principle – this allows reducing a simultaneous fixpoint to a sequence of individual fixpoints (see e.g. [1, Lemma 1.4.2]). The formula associated to the original automaton can now be read directly from the resulting equational system.

▶ **Theorem 28.** *For a parity (S,F)-automaton (A,α,Ω) with $A_{\max(\mathsf{ran}(\Omega))} = \{a\}$, there exists $\varphi_a \in \mu\mathcal{L}_\Lambda$ such that, for every $\mathsf{T}_S \circ F$-coalgebra (C,γ), if $e : [(e_n)_{n\in\mathsf{ran}(\Omega)}]$ is the extent of the product parity automaton between (C,γ) and (A,α,Ω) then $[\![\varphi_a]\!]_\gamma(c) = e(c,a)$ for $c \in C$.*

Proof (sketch). The proof closely follows that of [19, Theorem 34]. ◀

▶ Remark 29. The results of this section can easily be extended to the variant of $\mu\mathcal{L}_\Lambda$ described in Remark 6: while formula and model automata now have slightly different types, their product, itself a parity (S,F)-automaton, can be constructed in a similar way.

6 From Parity to Büchi Automata

We now present a direct reduction from *parity word automata* to *Büchi ones*. Parity word automata are parity $(S, F_{\Sigma,\Delta})$-automata, with $F_{\Sigma,\Delta} = \Sigma \times \mathsf{Id} + \Delta$ for alphabets Σ and Δ. Büchi automata have $\mathsf{ran}(\Omega) = \{1,2\}$. Here we additionally assume the semiring S to be total. Our reduction involves manipulating "*linear*" equational systems; their rhs*s* use operations that resemble matrix-vector multiplication and vector addition, as defined below.

▶ **Definition 30.** Let X and Y be sets, and let $M : X \times Y \to S$ and $v : Y \to S$ be a relation and a predicate respectively. Assume M and v to have finite support. $M \bullet v : X \to S$ is defined by $(M \bullet v)(x) = \sum_{y\in Y} M(x,y) \bullet v(y)$. For predicates $v_1, v_2 : Y \to S$, $v_1 + v_2$ is defined by pointwisely extending $+$ on S.

By the finite support property, $M \bullet v$ is well-defined even for X or Y infinite. The next lemma concerning linear equational systems is key to our reduction.

▶ **Lemma 31.** *Let a set $X = X_1 + \ldots + X_n$, a number $k \in [1,n]$ and a relation $M : X \times X \to S$ be fixed. Let $X_{\leq k} = X_1 + \ldots + X_k$ and $X_{>k} = X_{k+1} + \ldots + X_n$. For a predicate $v : X_{\leq k} \to S$, the equational systems $E^\eta_k(v)$ and $E^\mu_k(v)$, whose solutions belong to $S^{X_1} \times \ldots \times S^{X_k} \cong S^{X_{\leq k}}$, are defined as follows. Polarities $\eta_1, \ldots, \eta_k \in \{\mu,\nu\}$ of $E^\eta_k(v)$ can be chosen arbitrarily.*

$$E^\eta_k(v) = \begin{bmatrix} u_1 & =_{\eta_1} & M_1 \bullet [u_1,\ldots,u_k] + v_1 \\ & \vdots & \\ u_k & =_{\eta_k} & M_k \bullet [u_1,\ldots,u_k] + v_k \end{bmatrix},$$

$$E^\mu_k(v) = \begin{bmatrix} u_1 & =_{\mu} & M_1 \bullet [u_1,\ldots,u_k] + v_1 \\ & \vdots & \\ u_k & =_{\mu} & M_k \bullet [u_1,\ldots,u_k] + v_k \end{bmatrix}$$

Here each sub-relation $M_i : X_i \times X_{\leq k} \to S$ and each sub-predicate $v_i : X_i \to S$ are given by domain restriction. Then for any $v^\eta, v^\mu : X_{\leq k} \to S$, the solution of $E^\eta_k(v^\eta + v^\mu)$ is obtained by summing the solutions of $E^\eta_k(v^\eta)$ and $E^\mu_k(v^\mu)$.

The reduction proceeds in a step-wise manner, decrementing the largest parity, assumed without loss of generality to be even, by 2 at each step. An example is given in Figure 1. One reduction step is as follows:

1. Create a copy of states with parity lower than $n-1$. Incoming (only!) edges to those states are also copied. (This is only possible if the semiring is *total*.)
2. For the old copy of states with parity lower than $n-1$, let the new parity be $n-3$ and drop explicit terminations (i.e. transitions with a nullary letter).
3. Decrement the priorities of the other states by 2.

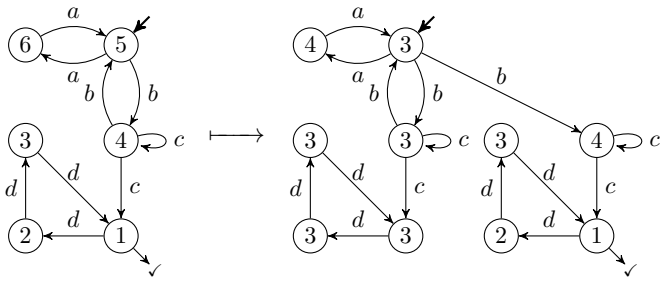


Figure 1 A step of the reduction from a parity automaton to a Büchi automaton. The number on each node denotes its parity. This step does not change the accepted language $((b\,c{*}\,b){*}\,(aa){*}){\omega} \mid ((b\,c{*}\,b){*}\,(aa){*}){*}\,b\,c{\omega} \mid ((b\,c{*}\,b){*}\,(aa){*}){*}\,b\,c{*}\,c(ddd){*}$.

The intuition is that, from old copies of states, a state with parity higher than $n-2$ must be visited again for acceptance; whereas from new copies, such a state must *not* be visited. The correctness of this translation is proved using Lemma 31. This also explains why the shape of F must be restricted to $F_{\Sigma,\Delta}$ – this makes extents be given by *linear* equational systems.

By repeatedly applying the reduction step, any parity $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}$ can be reduced to a Büchi $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}}$. Since the reduction is such that an accepting path of $\mathcal{A}$ corresponds to exactly *one* accepting path of $\mathcal{A}^{\mathrm{B}}$, idempotence of $+$ is not required.

▶ **Theorem 32.** *Let $\mathcal{A} = (A, \alpha, \Omega)$ be a parity $(S, F_{\Sigma,\Delta})$-automaton with $\mathsf{ran}(\Omega) = [1, n]$. Assume without loss of generality that n is even. The Büchi $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}} = (A^{\mathrm{B}}, \alpha^{\mathrm{B}}, \Omega^{\mathrm{B}})$ is given by*

$$A^{\mathrm{B}} = \coprod_{k\in[1,n]} A_k \times \{i \mid i \in [k, n], i \text{ is even}\}$$

$$\alpha^{\mathrm{B}}(a,i)(\iota_\sigma(a', j)) = \begin{cases} \alpha(a)(\iota_\sigma(a')) & \text{if } i = j, \text{ or } \lceil\Omega(a)\rceil = i \text{ and } i > j \\ 0 & \text{otherwise} \end{cases}$$

$$\alpha^{\mathrm{B}}(a,i)(\iota_\delta) = \begin{cases} \alpha(a)(\iota_\delta) & \text{if } \lceil\Omega(a)\rceil = i \\ 0 & \text{otherwise} \end{cases}$$

$$\Omega^{\mathrm{B}}(a,i) = \begin{cases} 2 & \text{if } \lceil\Omega(a)\rceil = i \text{ and } i \text{ is even} \\ 1 & \text{otherwise} \end{cases}$$

Here $\lceil\Omega(a)\rceil = \min\{k \mid k \geq \Omega(a) \text{ and } k \text{ is even}\}$. The automaton $\mathcal{A}^{\mathrm{B}}$ satisfies

$$\mathsf{ext}(\mathcal{A} \times \mathcal{M})(a, m) = \textstyle\sum_{(a,i)\in A^{\mathrm{B}}} \mathsf{ext}(\mathcal{A}^{\mathrm{B}} \times \mathcal{M})((a, i), m) \tag{5}$$

for any $\mathsf{T}_S \circ F_{\Sigma,\Delta}$-coalgebra $\mathcal{M}$ and each $(a, m) \in A \times M$.

▶ Remark 33. The definition of $\mathcal{A}^{\mathrm{B}}$ does not generalise to partial semirings S, due to the duplication of states involved. However, if S can be extended to a total semiring S' satisfying our assumptions, then carrying out the reduction for $\mathcal{A}$ as an $(S', F_{\Sigma,\Delta})$-automaton yields the desired result, namely a Büchi $(S', F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}}$ that satisfies condition (5). In particular, this means that one can treat the probabilistic case ($S = ([0, 1], +, 0, *, 1)$) by moving to the total semiring $S' = (\mathbb{R}^\infty, +, 0, *, 1)$. Specifically, one can model check formulas in our probabilistic linear time μ-calculus by (i) taking the product of the given model and formula automata, (ii) reducing the resulting automaton, viewed as an $(S', F_{\Sigma,\Delta})$-automaton, to a Büchi one, and (iii) computing the extent of the $(S', F_{\Sigma,\Delta})$-automaton thus obtained.

Input: parity automaton (A, α, Ω)
Output: extent $e : A \to S$ of (A, α, Ω)

1. **let** $e := [a \mapsto 0_S]$
2. $\mathit{Extent}(\max(\mathsf{ran}(\Omega)))$

Procedure $\mathit{Extent}(n \in \mathbb{N})$

1. **if** $n = 0$ **then** return **endif**
2. **for** $a \in A_n$ **do**
3. $\quad e(a) \leftarrow \begin{cases} 0_S, & \text{if } n \text{ is even} \\ 1_S, & \text{otherwise} \end{cases}$
4. **endfor**
5. **repeat**
6. $\quad$ **let** $\mathit{old} := e$
7. $\quad e \leftarrow \mathit{Extent}(n-1)$
8. $\quad$ **for** $a \in A_n$ **do**
9. $\quad\quad e(a) \leftarrow \sum_{i \in I} v_i \bullet \mathit{old}(a_1) \bullet \ldots \bullet \mathit{old}(a_{j_i})$
$\quad\quad\quad$ **where** $\alpha(a) = \sum_{i \in I} v_i \iota_\lambda(a_1, \ldots, a_{j_i})$
10. $\quad$ **endfor**
11. **until** $e = \mathit{old}$

- Lines 8-10 of the *Extent* procedure compute a better approximation of the extent for automaton states with the current parity n, based on a one-step unfolding of the automaton transition structure.
- Line 7 computes the extent of states with immediately lower parity, relative to the current values for states with parity n, through a recursive call to *Extent* (which may involve further recursive calls).
- Recursive calls to *Extent* update the same copy of e, and only make an additional copy to remember values from the previous step.

Figure 2 Algorithm for computing the extent $e : A \to S$ of (A, α, Ω).

Theorem 32 together with the existence of semantics preserving translations from $\mu\mathcal{L}_\Lambda$-formulas to parity (S, F)-automata and back (Theorems 24 and 28) now give us the following:

▶ **Corollary 34.** *For $F = F_{\Sigma,\Delta}$, the alternation degree 1 fragment of $\mu\mathcal{L}_\Lambda$ is fully expressive.*

7 An Algorithm for Computing Extents

We now describe an algorithm for computing the extent of a parity (S, F)-automaton, under the additional assumption that the length of any strictly ascending/descending chain in $(S, \sqsubseteq)$ is bounded. Both the boolean semiring and the bounded version of the tropical semiring (see Example 2) satisfy this assumption.

We fix a parity (S, F)-automaton (A, α, Ω). Thus, for $a \in A$, $\alpha(a)$ is a finite weighted sum of tuples $\iota_\lambda(a_1, \ldots, a_k)$ with $\lambda \in \Lambda$, $k = \mathsf{ar}(\lambda)$ and $a_i \in A$. The algorithm is described in Figure 2, and employs a recursive procedure $\mathit{Extent}(n \in \omega)$.

To see that the algorithm terminates, note that each call of $\mathit{Extent}(n)$ either increases or decreases all the values in A_n. The assumed bound on the length of strictly ascending/descending chains guarantees termination of each call: each iteration of the **repeat** ... **until** changes at least one of the values $e(a)$ with $a \in A_n$. Several improvements to the algorithm are possible, e.g. only remembering certain extent values (those with parity n) in the variable *old*; and deferring the recursive call until the approximation on the current parity saturates.

Next, we discuss complexity. If $|\mathsf{ran}(\Omega)| = 1$, and assuming for simplicity that S is finite, the algorithm has a time complexity which is quadratic in the size of the automaton – the length of a strictly increasing/decreasing chain in B^{A_i} is at most $|A_i||B|$, each iteration (lines 8-10 in the algorithm) increases/decreases at least one of the values $e(a)$, and takes time linear in $|A|$. If $|\mathsf{ran}(\Omega)| = m > 1$, then since each recursive call of *Extent* only updates values $e(a)$ with $a \in A_i$, for some $i \in \mathsf{ran}(\Omega)$, the time complexity is $O(|A|^{|\mathsf{ran}(\Omega)|}|B|^{|\mathsf{ran}(\Omega)|} \prod_{i \in \mathsf{ran}(\Omega)} |A_i|)$.

We conclude by noting that, for *Büchi* (S, F)-automata, the algorithm can be generalised to semirings S where only strictly *ascending* chains in $(S, \sqsubseteq)$ are required to have bounded length. This extends applicability e.g. to the tropical semiring. Instead of the exact extent of a

Büchi automaton, the generalised algorithm computes increasingly finer over-approximations of the extent. The boundedness assumption guarantees that *inner* calls to *Extent* terminate.

8 Related Work and Concluding Remarks

A translation from fixpoint calculi over an arbitrary signature Σ to Σ-*automata* is defined at an abstract level in [1, Section 7]. However, the fixpoint calculi of loc. cit. are intrinsically boolean (their semantics is given in terms of parity games), and thus do not subsume the quantitative logics described here. Our logics share many features with concrete μ-calculi, and our translations from formulas to automata and back resemble existing ones (see e.g. [19, Section 5.3]). Yet, our logics differ in their quantitative semantics, which in particular means that a semantics based on parity games is not available anymore.

Our translation from formulas to automata exposes and exploits the coalgebraic structures present in both models and formulas. Theorems 24 and 28 are thus similar to results in [18], which also provide a coalgebraic perspective on the connection between fixpoint logics and automata. Differently from [18], our construction of the automaton for a formula avoids the use of silent transitions and, more importantly, applies also to *quantitative* logics.

Our results go beyond existing ones, both in the case of probabilistic systems (given our choice of logics) and in the case of weighted systems (where we are not aware of similar translations). In particular, for probabilistic systems, our logics are subtly different from existing ones (they contain sub-convex combinations of formulas but no pure disjunctions or conjunctions), yet are at least as expressive (see Remark 6).

Our parity to Büchi translation (Theorem 32) is novel in two ways: (i) unlike existing translations (e.g. from probabilistic Rabin to probabilistic Büchi automata [2]) which preserve only the *qualitative* language, our translation preserves the *quantitative* language, and (ii) our result comes with a proof which is not a generalisation of any proof we are aware of in the qualitative case. Thus, the jump from a qualitative to a quantitative setting is non-trivial.

Our admittedly trivial model checking algorithm has complexity similar to that of known algorithms for the qualitative case [14], while also being applicable in quantitative settings. Currently this only includes semirings with *finite* strictly ascending/descending chains. Such semirings can e.g. be used to model resource usage with bounded resources. The study of more generally applicable algorithms is left as future work. For this, our recent lattice-based generalisation of the notion of progress measure [13] is expected to prove useful.

Acknowledgments. We thank the anonymous reviewers for their insightful comments.

References

1 A. Arnold and D. Niwiński. *Rudiments of μ-Calculus*. Studies in Logic and the Foundations of Mathematics. North-Holland, 2001.

2 C. Baier, M. Größer, and N. Bertrand. Probabilistic ω-automata. *J. ACM*, 59(1):1, 2012.

3 C. Baier and J.-P. Katoen. *Principles of model checking*. MIT Press, 2008.

4 C. Cîrstea. A coalgebraic approach to linear-time logics. In A. Muscholl, editor, *Foundations of Software Science and Computation Structures, 17th International Conference*, pages 426–440. Springer, 2014.

5 C. Cîrstea. A coalgebraic approach to quantitative linear time logics. *CoRR*, abs/1612.07844, 2016. URL: `http://arxiv.org/abs/1612.07844`.

6 C. Cîrstea. From branching to linear time, coalgebraically. *Fundamenta Informaticae*, 150:1–28, 2017.

7 R. Cleaveland, M. Klein, and B. Steffen. Faster model checking for the modal μ-calculus. In *Computer Aided Verification, 4th International Workshop*, pages 410–422. Springer, 1993.

8 D. Coumans and B. Jacobs. Scalars, monads, and categories. In *Quantum Physics and Linguistics. A Compositional, Diagrammatic Discourse*, pages 184–216. Oxford Univ. Press, 2013.

9 M. Dam. Fixed points of Büchi automata. In R. Shyamasundar, editor, *Foundations of Software Technology and Theoretical Computer Science, 12th Conference*, pages 39–50. Springer, 1992.

10 B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order (2. ed.)*. Cambridge University Press, 2002.

11 B. Farwer. ω-Automata. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games: A Guide to Current Research*, pages 3–20. Springer, 2002.

12 M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194 – 211, 1979.

13 I. Hasuo, S. Shimizu, and C. Cîrstea. Lattice-theoretic progress measures and coalgebraic model checking. In *43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 718–732, 2016.

14 D. Kirsten. Alternating tree automata and parity games. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata Logics, and Infinite Games: A Guide to Current Research*, pages 153–167. Springer, 2002.

15 I. Meinecke. A weighted μ-calculus on words. In *13th International Conference on Developments in Language Theory*, pages 384–395. Springer, 2009.

16 M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

17 N. Urabe, S. Shimizu, and I. Hasuo. Coalgebraic trace semantics for büchi and parity automata. In J. Desharnais and R. Jagadeesan, editors, *27th International Conference on Concurrency Theory*, volume 59 of *LIPICS*, pages 24:1–24:15, 2016.

18 Y. Venema. Lectures on the modal μ-calculus. Lecture notes, Institute for Logic, Language and Computation, University of Amsterdam, 2012.

19 I. Walukiewicz. Automata and logic, 2001. Notes available from `http://www.labri.fr/perso/igw/Papers/igw-eefss01.ps`.

Automata Minimization: a Functorial Approach*†

Thomas Colcombet[1] and Daniela Petrişan[2]

1 **CNRS, IRIF, Univ. Paris-Diderot, Paris 7, France**
`thomas.colcombet@irif.fr`
2 **CNRS, IRIF, Univ. Paris-Diderot, Paris 7, France**
`petrisan@irif.fr`

Abstract

In this paper we regard languages and their acceptors – such as deterministic or weighted automata, transducers, or monoids – as functors from input categories that specify the type of the languages and of the machines to categories that specify the type of outputs.

Our results are as follows: a) We provide sufficient conditions on the output category so that minimization of the corresponding automata is guaranteed. b) We show how to lift adjunctions between the categories for output values to adjunctions between categories of automata. c) We show how this framework can be applied to several phenomena in automata theory, starting with determinization and minimization (previously studied from a coalgebraic and duality theoretic perspective). We apply in particular these techniques to Choffrut's minimization algorithm for subsequential transducers and revisit Brzozowski's minimization algorithm.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases functor automata, minimization, Choffrut's minimization algorithm, subsequential transducers, Brzozowski's minimization algorithm

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.8

1 Introduction

There is a long tradition of interpreting results of automata theory using the lens of category theory. Typical instances of this scheme interpret automata as algebras (together with a final map) as put forward in [3, 14, 1], or as coalgebras (together with an initial map), see for example [16]. This dual narrative proved very useful [7] in explaining at an abstract level Brzozowski's minimization algorithm and the duality between reachability and observability (which goes back all the way to the work of Arbib, Manes and Kalman).

In this paper, we adopt a slightly different approach, and we define directly the notion of an automaton (over finite words) as a functor from a category representing input words, to a category representing the computation and output spaces. The notions of a language and of a language accepted by an automaton are adapted along the same pattern.

We provide several developments around this idea. First, we recall (see [12]) that the existence of a minimal automaton for a language is guaranteed by the existence of an initial and a final automaton in combination with a factorization system. Additionally, we explain how, in the functor presentation that we have adopted, the existence of initial and final

* This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No.670624), and by the DeLTA ANR project (ANR-16-CE40-0007). The authors also thank the Simons Institute for the Theory of Computing where this work has been partly developed.

† A knowledge enriched version of the paper is available at `https://arxiv.org/abs/1711.03063`.

automata for a language can be phrased in terms of Kan extensions. We also show how adjunctions between categories can be lifted to the level of automata for a language in these categories (Lemma 8). This lifting accounts for several constructions in automata theory, determinization to start with. We then use this framework in the explanation of two well-known constructions in automata theory.

The most involved contribution (Theorem 12) is to rephrase the minimization result of Choffrut for subsequential transducers in this framework. We do this by instantiating the category of outputs with the Kleisli category for the monad $TX = B^* \times X + 1$, where B is the output alphabet of the transducers. In this case, despite the lack of completeness of the ambient category, one can still prove the existence of an initial and final automaton, as well as, surprisingly, of a factorization system.

The second concrete application is a proof of correctness of Brzozowski's minimization algorithm. Indeed, determinization of automata can be understood as lifting the Kleisli adjunction between the categories Rel (of sets and relations) and Set (of sets and functions); and reversing a nondeterministic automaton can be understood as lifting the self-duality of Rel. In Section 5 we show how Brzozowski's minimization algorithm can be explained by combining several liftings of adjunctions, and in particular as lifting the adjunction between Set and its opposite category Set^{op}, thus recovering results from [7].

Related work. Many of the constructions outlined here have already been explained from a category-theoretic perspective, using various techniques. For example, the relationship between minimization and duality was subject to numerous papers, see [6, 7, 8] and the references therein. The coalgebraic perspective on minimization was also emphasised in papers such as [2, 19]. Understanding determinization and codeterminization, as well as studying trace semantics via lifting of adjunctions to coalgebras was considered in [17, 18], and is related to our results from Section 5.2. Subsequential transducers were understood coalgebraically in [15].

The paper which is closest in spirit to our work is a seemingly forgotten paper [4]. However, in this work, Bainbridge models the *state space* of the machines as a functor. Left and right Kan extensions are featured in connection with the initial and final automata, but in a slightly different setting. Lemma 8, which albeit technically simple, has surprisingly many applications, builds directly on his work.

2 Languages and Automata as Functors

In this section, we introduce the notion of automata via functors, which is the common denominator of the different contributions of the paper. We introduce this definition starting from the special case of classical deterministic automata.

In the standard definition, a deterministic automaton is a tuple:

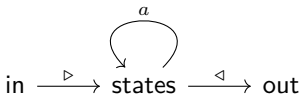
$$\langle Q, A, q_0, F, \delta \rangle$$

where Q is a set of states, A is an alphabet (not necessarily finite), $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and $\delta_a \colon Q \to Q$ is the transition map for all letters $a \in A$. The semantics of an automaton is given by defining what is a run over an input word $u \in A^*$, and whether it is accepting or not. Given a word $e = a_1 \dots a_n$, the automaton accepts the word if $\delta_{a_n} \circ \cdots \circ \delta_{a_1}(q_0) \in F$, and otherwise rejects it.

If we see q_0 as a map *init* from the one element set $1 = \{0\}$ to Q, that maps 0 to q_0, and F as a map *final* from Q to the set $2 = \{0, 1\}$, where 1 means 'accept' and 0 means

'reject', then the semantics of the automaton is to associate to each word $u = a_1 \dots a_n$ the map from 1 to 2 defined as $\mathit{final} \circ \delta_{a_n} \circ \dots \circ \delta_{a_1} \circ \mathit{init}$. If this map is (constant equal to) 1, this means that the word is accepted, and otherwise it is rejected.

Pushing this idea further, we can see the semantics of the automaton as a functor from the free category generated by the graph on the right to Set, and more precisely one that sends the object in to 1 and out to 2. In the above category, the arrows from in to out are of the form $\triangleright w \triangleleft$ for w an arbitrary word in A^*.



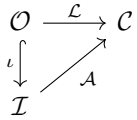
Furthermore, since a language can be seen as a map from A^* to $1 \to 2$, we can model it as a functor from the full subcategory on objects in and out to the category Set, which maps in to 1 and out to 2.

In this section we fix an arbitrary small category $\mathcal{I}$ and a full subcategory $\mathcal{O}$. We denote by ι the inclusion functor

$$\mathcal{O} \xhookrightarrow{\iota} \mathcal{I}\,.$$

We think of $\mathcal{I}$ as a specification of the inner computations that an automaton can perform, including black box behaviour, not observable from the outside. On the other hand, the full subcategory $\mathcal{O}$ specifies the observable behaviour of the automaton, that is, the language it accepts. In this interpretation, a machine/automaton $\mathcal{A}$ is a functor from $\mathcal{I}$ to a category of outputs $\mathcal{C}$, and the "behaviour" or "language" of $\mathcal{A}$ is the functor $\mathcal{L}(\mathcal{A})$ obtained by precomposition with the inclusion $\mathcal{O} \xhookrightarrow{\iota} \mathcal{I}$. We obtain the following definition:

▶ **Definition 1** (languages and the categories of automata for them)**.**
A $\mathcal{C}$-language is a functor $\mathcal{L}\colon \mathcal{O} \to \mathcal{C}$ and a $\mathcal{C}$-automaton is a functor $\mathcal{A}\colon \mathcal{I} \to \mathcal{C}$. A $\mathcal{C}$-automaton $\mathcal{A}$ accepts a $\mathcal{C}$-language $\mathcal{L}$ when $\mathcal{A} \circ \iota = \mathcal{L}$; i.e. the diagram below commutes:



We write $\mathsf{Auto}(\mathcal{L})$ for the subcategory of the functor category $[\mathcal{I}, \mathcal{C}]$ where

1. objects are $\mathcal{C}$-automata that accept $\mathcal{L}$.
2. arrows are natural transformations $\alpha\colon \mathcal{A} \to \mathcal{B}$ so that the natural transformation obtained by composition with the inclusion functor ι is the identity natural transfomation on $\mathcal{L}$, that is, $\alpha \circ \iota = id_{\mathcal{L}}$.

2.1 Minimization of $\mathcal{C}$-automata

In this section we show that the notion of a minimal automaton is an instance of a more generic notion of minimal object that can be defined in an arbitrary category $\mathcal{K}$ whenever there exist an initial object, a final object, and a factorization system $(\mathcal{E}, \mathcal{M})$.

Let X, Y be two objects of $\mathcal{K}$. We say that:

$$X \quad (\mathcal{E},\mathcal{M})\text{-divides} \quad Y \qquad \text{if} \qquad X \text{ is an } \mathcal{E}\text{-quotient of an } \mathcal{M}\text{-subobject of } Y.$$

Let us note immediately that in general this notion of $(\mathcal{E},\mathcal{M})$-divisibility may not be transitive[1]. It is now natural to define an object M to be $(\mathcal{E},\mathcal{M})$-minimal in the category, if it $(\mathcal{E},\mathcal{M})$-divides all objects of the category. Note that there is no reason a priori that an $(\mathcal{E},\mathcal{M})$-minimal object in a category, if it exists, be unique up to isomorphism. Nevertheless, in our case, when the category has both initial and a final object, we can state the following minimization lemma:

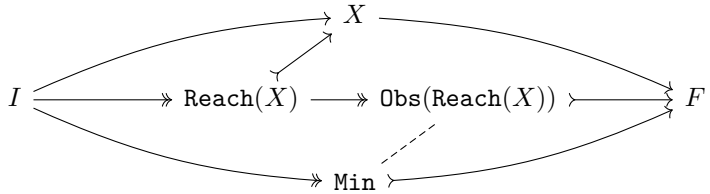
▶ **Lemma 2.** *Let $\mathcal{K}$ be a category with initial object I and final object F and let $(\mathcal{E},\mathcal{M})$ be a factorization system for $\mathcal{K}$. Define for every object X:*

- $\mathtt{Min}$ *to be the factorization of the only arrow from I to F,*
- $\mathtt{Reach}(X)$ *to be the factorization of the only arrow from I to X, and* $\mathtt{Obs}(X)$ *to be the factorization of the only arrow from X to F.*

Then

- $\mathtt{Min}$ *is $(\mathcal{E},\mathcal{M})$-minimal, and*
- $\mathtt{Min}$ *is isomorphic to both* $\mathtt{Obs}(\mathtt{Reach}(X))$ *and* $\mathtt{Reach}(\mathtt{Obs}(X))$ *for all objects X.*

Proof. The proof essentially consists of a diagram:

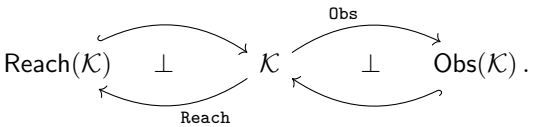


Using the definition of $\mathtt{Reach}$ and $\mathtt{Obs}$, and the fact that $\mathcal{E}$ is closed under composition, we obtain that $\mathtt{Obs}(\mathtt{Reach}(X))$ is an $(\mathcal{E},\mathcal{M})$-factorization of the only arrow from I to F. Thus, thanks to the diagonal property of a factorization system, $\mathtt{Min}$ and $\mathtt{Obs}(\mathtt{Reach}(X))$ are isomorphic. Hence, furthermore, since $\mathtt{Obs}(\mathtt{Reach}(X))$ $(\mathcal{E},\mathcal{M})$-divides X by construction, the same holds for $\mathtt{Min}$. In a symmetric way, $\mathtt{Reach}(\mathtt{Obs}(X))$ is also isomorphic to $\mathtt{Min}$. ◀

An object X of $\mathcal{K}$ is called reachable when X is isomorphic to $\mathtt{Reach}(X)$. We denote by $\mathsf{Reach}(\mathcal{K})$ the full subcategory of $\mathcal{K}$ consisting of reachable objects. Similarly, an object X of $\mathcal{K}$ is called observable when X is isomorphic to $\mathtt{Obs}(X)$. We denote by $\mathsf{Obs}(\mathcal{K})$ the full subcategory of $\mathcal{K}$ consisting of observable objects.

We can express reachability $\mathtt{Reach}$ and observability $\mathtt{Obs}$ as the right, respectively the left adjoint to the inclusion of $\mathsf{Reach}(\mathcal{K})$, respectively of $\mathsf{Obs}(\mathcal{K})$ into $\mathcal{K}$. It is indeed a standard fact that factorization systems give rise to reflective subcategories, see [9]. In our case, this is the reflective subcategory $\mathsf{Obs}(\mathcal{K})$ of $\mathcal{K}$. By a dual argument, the category $\mathsf{Reach}(\mathcal{K})$ is coreflective in $\mathcal{K}$. We can summarize these facts in the next lemma.

▶ **Lemma 3.** *Let $\mathcal{K}$ be a category with initial object I and final object F and let $(\mathcal{E},\mathcal{M})$ be a factorization system for $\mathcal{K}$. We have the adjunctions*



[1] There are nevertheless many situations for which it is the case; In particular when the category is regular, and $\mathcal{E}$ happens to be the class of regular epis. This covers in particular the case of all algebraic categories with $\mathcal{E}$-quotients being the standard quotients of algebras, and $\mathcal{M}$-subobjects being the standard subalgebras.

In what follows we will instantiate $\mathcal{K}$ with the category $\mathsf{Auto}(\mathcal{L})$ of $\mathcal{C}$-automata accepting a language $\mathcal{L}$. Assuming the existence of an initial and final automaton for $\mathcal{L}$ – denoted by $\mathcal{A}^{init}(\mathcal{L})$, respectively $\mathcal{A}^{final}(\mathcal{L})$ – and, of a factorization system, we obtain the functorial version of the usual notions of reachable sub-automaton $\mathtt{Reach}(\mathcal{A})$ and observable quotient automaton $\mathtt{Obs}(\mathcal{A})$ of an automaton $\mathcal{A}$. The minimal automaton $\mathtt{Min}(\mathcal{L})$ for the language $\mathcal{L}$ is obtained via the factorization

$$\mathcal{A}^{init}(\mathcal{L}) \twoheadrightarrow \mathtt{Min}(\mathcal{L}) \rightarrowtail \mathcal{A}^{final}(\mathcal{L})\,.$$

Lemma 2 implies that the minimal automaton divides any other automaton recognising the language, while Lemma 3 instantiates to the results of [7, Section 9.4].

2.2 Minimization of $\mathcal{C}$-automata: sufficient conditions on $\mathcal{C}$

We now can list sufficient conditions on $\mathcal{C}$ so that the category of $\mathcal{C}$-automata $\mathsf{Auto}(\mathcal{L})$ accepting a $\mathcal{C}$-language $\mathcal{L}$ satisfies the three conditions of Lemma 2.

We start with the factorization system. It is well known that given a factorization system $(\mathcal{E},\mathcal{M})$ on $\mathcal{C}$, we can extend it to a factorization system $(\mathcal{E}_{[\mathcal{I},\mathcal{C}]},\mathcal{M}_{[\mathcal{I},\mathcal{C}]})$ on the functor category $[\mathcal{I},\mathcal{C}]$ in a pointwise fashion. That is a natural transformation is in $\mathcal{E}_{[\mathcal{I},\mathcal{C}]}$ if all its components are in $\mathcal{E}$, and analogously, a natural transformation is in $\mathcal{M}_{[\mathcal{I},\mathcal{C}]}$ if all its components are in $\mathcal{M}$. In turn, the factorization system on the functor category $[\mathcal{I},\mathcal{C}]$ induces a factorization system on each subcategory $\mathsf{Auto}(\mathcal{L})$.

▶ **Lemma 4.** *If $\mathcal{C}$ has a factorization system $(\mathcal{E},\mathcal{M})$, then $\mathsf{Auto}(\mathcal{L})$ has a factorization system $(\mathcal{E}_{\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})},\mathcal{M}_{\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})})$, where $\mathcal{E}_{\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})}$ consists of all the natural transformations with components in $\mathcal{E}$ and $\mathcal{M}_{\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})}$ consists of all natural transformations with components in $\mathcal{M}$.*

The proof of Lemma 4 is the same as the classical one that shows that factorization systems can be lifted to functor categories.

▶ **Lemma 5.** *If the left Kan extension $\mathsf{Lan}_\iota\mathcal{L}$ of $\mathcal{L}$ along ι exists, then it is an initial object in $\mathsf{Auto}(\mathcal{L})$, that is, $\mathcal{A}^{init}(\mathcal{L})$ exists and is isomorphic to $\mathsf{Lan}_\iota\mathcal{L}$.*

Dually, if the right Kan extension $\mathsf{Ran}_\iota\mathcal{L}$ of $\mathcal{L}$ along ι exists, then so does the final object $\mathcal{A}^{final}(\mathcal{L})$ of $\mathsf{Auto}(\mathcal{L})$ and $\mathcal{A}^{final}(\mathcal{L})$ is isomorphic to $\mathsf{Ran}_\iota\mathcal{L}$.

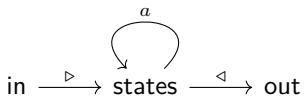
Proof Sketch. Assume the left Kan extension exists. Then the canonical natural transformation $\mathcal{L} \to \mathsf{Lan}_\iota\mathcal{L} \circ \iota$ is an isomorphism since ι is full and faithful. Whenever $\mathcal{A}$ accepts $\mathcal{L}$, that is, $\mathcal{A} \circ \iota = \mathcal{L}$, we obtain the required unique morphism $\mathsf{Lan}_\iota\mathcal{L} \to \mathcal{A}$ using the universal property of the Kan extension. The argument for the right Kan extension follows by duality. ◀

▶ **Corollary 6.** *Assume $\mathcal{C}$ is complete, cocomplete and has a factorization system and let $\mathcal{L}$ be a $\mathcal{C}$-language. Then the initial $\mathcal{L}$-automaton and the final $\mathcal{L}$-automaton exist and are given by the left, respectively right Kan extensions of $\mathcal{L}$ along ι. Furthermore, the minimal $\mathcal{C}$-automaton $\mathtt{Min}(\mathcal{L})$ accepting $\mathcal{L}$ is obtained via the factorization $\mathsf{Lan}_\iota\mathcal{L} \twoheadrightarrow \mathtt{Min}(\mathcal{L}) \rightarrowtail \mathsf{Ran}_\iota\mathcal{L}$.*

▶ Remark. Depending on the category $\mathcal{I}$, we may relax the conditions in Corollary 6, see Lemma 7. Furthermore, we emphasise that these conditions are only sufficient. In Section 4 we consider the example of subsequential transducers and we instantiate $\mathcal{C}$ with a Kleisli category. Although this category does not have powers, the final automaton exists.

3 Word Automata

Hereafter, we restrict our attention to the case of word automata, for which the input category $\mathcal{I}$ is the three-object category with arrows spanned by $\triangleright$, $\triangleleft$ and a for all $a \in A$, as in the diagram below and where the composite of $\mathsf{states} \xrightarrow{w} \mathsf{states} \xrightarrow{w'} \mathsf{states}$ is given by the concatenation ww'.

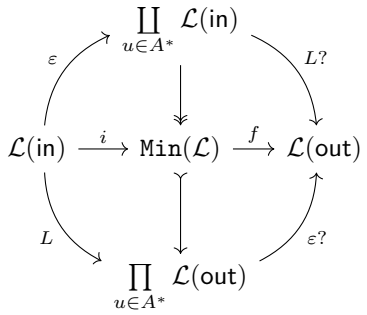


Let $\mathcal{O}$ be the full subcategory of $\mathcal{I}$ on objects in and out. We consider $\mathcal{C}$-languages, which are now functors $\mathcal{L}\colon \mathcal{O} \to \mathcal{C}$. If $\mathcal{L}(\mathsf{in}) = X$ and $\mathcal{L}(\mathsf{out}) = Y$ we call $\mathcal{L}$ a $(\mathcal{C}, X, Y)$-language. Similarily, we consider $\mathcal{C}$-automata that are functors $\mathcal{A}\colon \mathcal{I} \to \mathcal{C}$. If $\mathcal{A}(\mathsf{in}) = X$ and $\mathcal{A}(\mathsf{out}) = Y$ we call $\mathcal{A}$ a $(\mathcal{C}, X, Y)$-automaton.

The next lemma refines Corollary 6. It appeared in [5] in the $(\mathsf{Set}, 1, 2)$-language case.

▶ **Lemma 7** (from [12]). *If $\mathcal{C}$ has countable products and countable coproducts, and a factorization system, then the minimal $\mathcal{C}$-automaton accepting $\mathcal{L}$ is obtained via the factorization in the next diagram.*

The initial automaton has as state space the copower $\coprod_{u \in A^*} \mathcal{L}(\mathsf{in})$. In [12] we gave a direct proof of initiality, but here we can also notice that this is exactly what the colimit computation of the left Kan extension of $\mathcal{L}$ along ι yields – using the fact that there are no morphisms from out to states in $\mathcal{I}$ and the only morphism on which you take the colimit are of the form $\triangleright w\colon \mathsf{in} \to \mathsf{states}$ for all $w \in A^*$.



3.1 Lifting Adjunctions to Categories of Automata

In this section we will juggle with languages and automata interpreted over different categories connected via adjunctions.

Assume we have an adjunction between two categories $\mathcal{C}$ and $\mathcal{D}$

$$\mathcal{C} \underset{G}{\overset{F}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathcal{D},$$

with $F \dashv G\colon \mathcal{D} \to \mathcal{C}$. Let $(-)^*$ and $(-)_*$ denote the induced natural isomorphisms between the homsets. In particular, given objects I in $\mathcal{C}$ and O in $\mathcal{D}$, we have bijections

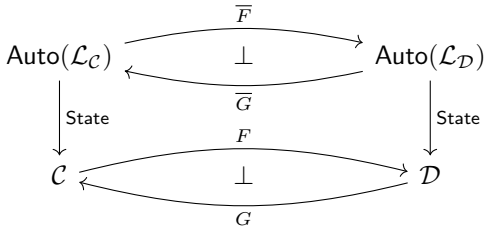
$$\mathcal{C}(I, GO) \underset{(-)_*}{\overset{(-)^*}{\rightleftarrows}} \mathcal{D}(FI, O) \tag{1}$$

These bijections induce a one-to-one correspondence between $(\mathcal{C}, I, GO)$-languages and $(\mathcal{D}, FI, O)$-languages, which by an abuse of notation we denote by the same symbols:

$$(\mathcal{C}, I, GO)\text{-languages} \underset{(-)_*}{\overset{(-)^*}{\rightleftarrows}} (\mathcal{D}, FI, O)\text{-languages}$$

Indeed, given a $(\mathcal{C}, I, GO)$-language $\mathcal{L} \colon \mathcal{O} \to \mathcal{C}$ we obtain a $(\mathcal{D}, FI, O)$-language $\mathcal{L}^* \colon \mathcal{O} \to \mathcal{D}$ by setting $\mathcal{L}^*(\triangleright w \triangleleft) = (\mathcal{L}(\triangleright w \triangleleft))^* \in \mathcal{D}(FI, O)$. Conversely, given a $(\mathcal{D}, FI, O)$-language $\mathcal{L}'$ we obtain a $(\mathcal{C}, I, GO)$-language $(\mathcal{L}')_*$ by setting $(\mathcal{L}')_*(\triangleright w \triangleleft) = (\mathcal{L}'(\triangleright w \triangleleft))_*$.

▶ **Lemma 8.** *Assume $\mathcal{L}_\mathcal{C}$ and $\mathcal{L}_\mathcal{D}$ are $(\mathcal{C}, I, GO)$-, respectively $(\mathcal{D}, FI, O)$-languages so that $\mathcal{L}_\mathcal{D} = (\mathcal{L}_\mathcal{C})^*$. Then the adjunction $F \dashv G$ lifts to an adjunction $\overline{F} \dashv \overline{G} \colon \mathsf{Auto}(\mathcal{L}_\mathcal{D}) \to \mathsf{Auto}(\mathcal{L}_\mathcal{C})$. The lifted functors $\overline{F}$ and $\overline{G}$ are defined as F, resp. G on the state object, that is, the following diagram commutes*



where the functor $\mathsf{State} \colon \mathsf{Auto}(\mathcal{L}_\mathcal{C}) \to \mathcal{C}$ *is the evaluation at* states*, that is, it sends an automaton $\mathcal{A} \colon \mathcal{I} \to \mathcal{C}$ to $\mathcal{A}(\mathsf{states})$.*

Proof sketch. The functor $\overline{F}$ maps an automaton $\mathcal{A} \colon \mathcal{I} \to \mathcal{C}$ from $\mathsf{Auto}(\mathcal{L}_\mathcal{C})$ to the $\mathcal{D}$-automaton $\overline{F}\mathcal{A} \colon \mathcal{I} \to \mathcal{D}$ mapping $\triangleright \colon \mathsf{in} \to \mathsf{states}$ to $F\mathcal{A}(\triangleright)$, $a \colon \mathsf{states} \to \mathsf{states}$ to $F(\mathcal{A}(a))$ and $\triangleleft \colon \mathsf{states} \to \mathsf{out}$ to the adjoint transpose $(\mathcal{A}(\triangleleft))^*$ of $\mathcal{A}(\triangleleft)$. The functor $\overline{G}$ is defined similarly. ◀

4 Choffrut's minimization of subsequential transducers

In [10, 11] Choffrut establishes a minimality result for subsequential transducers, which are deterministic automata that output a word while processing their input. In this section, we show the existence of minimal subsequential transducers using our functorial framework.

We first present the model of subsequential transducers in Section 4.1, show how these can be identified with automata in the Kleisli category of a suitably chosen monad, and state the minimization result, Theorem 12. The subsequent sections provide the necessary material for proving the theorem.

4.1 Subsequential transducers and automata in a Kleisli category

Subsequential transducers are (finite state) machines that compute partial functions from input words in some alphabet A to output words in some other alphabet B. In this section, we recall the classical definition of these objects, and show how it can be phrased categorically.

▶ **Definition 9.** A subsequential transducer is a tuple

$$T = (Q, A, B, q_0, t, u_0, (- \cdot a)_{a \in A}, (- * a)_{a \in A}) \ ,$$

where

- A is the input alphabet and B the output one,
- Q is a (finite) set of states.
- q_0 is either undefined or belongs to Q and is called the initial state of the transducer.
- $t\colon Q \rightharpoonup B^*$ is a partial termination function.
- $u_0 \in B^*$ is defined if and only if q_0 is, and is the initialization value.
- $- \cdot a\colon Q \rightharpoonup Q$ is the partial transition function for the letter a, for all $a \in A$.
- $- * a\colon Q \rightharpoonup B^*$ is the partial production function for the letter a for all $a \in A$; it is required that $q * a$ be defined if and only if $(q \cdot a)$ is.

A subsequential transducer T computes a partial function $[\![T]\!]\colon A^* \rightharpoonup B^*$ defined as:

$$[\![T]\!](a_1 \dots a_n) = u_0(q_0 * a_1)(q_1 * a_2) \dots (q_{n-1} * a_n)t(q_n) \qquad \text{for all } a_1 \dots a_n \in A^*,$$

where for each $1 \leq i \leq n$ either q_i is undefined or belongs to Q and is given by $q_i = q_{i-1} \cdot a_i$. Furthermore, $[\![T]\!](a_1 \dots a_n)$ is undefined when at least one of $q_0, \dots, q_n$ or $t(q_n)$ is so.

These subsequential transducers are modeled in our framework as automata in the category of free algebras for the monad $\mathcal{T}$, that we describe now.

▶ **Definition 10.** The monad $\mathcal{T}\colon \mathsf{Set} \to \mathsf{Set}$ is defined by

$$\mathcal{T}(X) = B^* \times X + 1$$

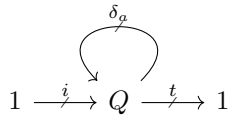
with unit η_X and multiplication μ_X defined for all $x \in X$ and $w, u \in B^*$ as:

$$\eta_X\colon \begin{array}{l} X \to B^* \times X + 1 \\ x \mapsto (\varepsilon, x) \end{array} \qquad \mu_X\colon \begin{array}{r} \mathcal{T}^2 X \to \mathcal{T} X \\ (w, (u, x)) \mapsto (wu, x) \\ (w, \perp) \mapsto \perp \\ \perp \mapsto \perp \end{array}$$

where we denote by $\perp$ the unique element of 1 (used to model the partiality of functions).

Recall that the category of free $\mathcal{T}$-algebras is the Kleisli category for $\mathcal{T}$, $\mathsf{Kl}(\mathcal{T})$, that has as objects sets $X, Y, \dots$ and as morphisms $f\colon X \nrightarrow Y$ functions $f\colon X \to B^* \times Y + 1$ in Set, that is a partial function from X to $B^* \times Y$.

Let T be a subsequential transducer. The initial state of the transducer q_0 and the initialization value u_0 together form a morphism $i\colon 1 \nrightarrow Q$ in the category $\mathsf{Kl}(\mathcal{T})$. Similarly, the partial transition function and the partial production function for a letter a of the input alphabet A are naturally identified to Kleisli morphisms $\delta_a\colon Q \nrightarrow Q$ in $\mathsf{Kl}(\mathcal{T})$. Finally, the partial termination function together with the partial production function are nothing but a Kleisli morphism of the form $t\colon Q \nrightarrow 1$. To summarise, we obtained that a subsequential transducer T in the sense of [11] is specified by the following morphisms in $\mathsf{Kl}(\mathcal{T})$:



that is by a functor $\mathcal{A}_T\colon \mathcal{I} \to \mathsf{Kl}(\mathcal{T})$ or equivalently, a $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automaton. The subsequential function realised by the transducer T is a partial function $A^* \rightharpoonup B^*$ and is fully captured by the $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-language $\mathcal{L}_T\colon \mathcal{O} \to \mathsf{Kl}(\mathcal{T})$ accepted by $\mathcal{A}_T$, which is obtained as $\mathcal{A}_T \circ \iota$. Indeed, this $\mathsf{Kl}(\mathcal{T})$-language gives for each word $w \in A^*$ a Kleisli morphism

$\mathcal{L}_T(\triangleright w \triangleleft) \colon 1 \nrightarrow 1$, or equivalently, outputs for each word in A^* either a word in B^* or the undefined element $\bot$.

Putting all this together, we can state the following lemma, which validates the categorical encoding of subsequential transducers:

▶ **Lemma 11.** *Subsequential transducers are in one to one correspondence with* $(\mathsf{Kl}(\mathcal{T}), 1, 1)$*-automata, and partial maps from* A^* *to* B^* *are in one to one correspondence with* $(\mathsf{Kl}(\mathcal{T}), 1, 1)$*-languages. Furthermore, the acceptance of languages is preserved under these bijections.*

In the rest of this section we will see how to obtain Choffrut's minimization result as an application of Lemma 2. I.e., we have to provide in the category of $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automata,

1. an initial object,
2. a final object, and,
3. a factorization system.

The existence of the initial transducer is addressed in Section 4.3, the one of the final transducer is the subject of Section 4.4. In Section 4.5 we show how to construct a factorization system@factorization transducer. Together, we obtain:

▶ **Theorem 12** (Categorical version of [10, 11])**.** *For every* $(\mathsf{Kl}(\mathcal{T}), 1, 1)$*-language, there exists a minimal* $(\mathsf{Kl}(\mathcal{T}), 1, 1)$*-automaton for it.*

Let us note that only the existence of the automaton is mentioned in this statement, and the way to compute it effectively is not addressed as opposed to Choffrut's work. Nevertheless, Lemma 2 describes what are the basic functions that have to be implemented, namely `Reach` and `Obs`.

The rest of this section is devoted to establishing the three above mentioned points. Unfortunately, as it is usually the case with Kleisli categories, $\mathsf{Kl}(\mathcal{T})$ is neither complete, nor cocomplete. It does not even have binary products, let alone countable powers. Also, the existence of a factorization system does not generally hold in Kleisli categories. Hence, providing the above three pieces of information requires a bit of work.

In the next section we present an adjunction between categories of $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automata and $(\mathsf{Set}, 1, B^*)$-automata which is then used in the subsequent ones for proving the existence of initial and final automata. We finish the proof with a presentation of the factorization systems.

4.2 Back and forth to automata in Set

In order to understand what are the properties of the category of $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automata, an important tool will be the ability to see alternatively a subsequential transducer as an automaton in $\mathsf{Kl}(\mathcal{T})$ as described above, or as an automaton in Set, since Set is much better behaved than $\mathsf{Kl}(\mathcal{T})$. These two points of view are related through an adjunction, making use of the results of Section 3.1 and Lemma 4.

Indeed, we start from the well known adjunction between Set and $\mathsf{Kl}(\mathcal{T})$:

$$\mathsf{Set} \underset{U_{\mathcal{T}}}{\overset{F_{\mathcal{T}}}{\underset{\bot}{\rightleftarrows}}} \mathsf{Kl}(\mathcal{T})\,. \tag{2}$$

We recall that the free functor $F_{\mathcal{T}}$ is defined as the identity on objects, while for any function $f \colon X \to Y$ the morphism $F_{\mathcal{T}} f \colon X \nrightarrow Y$ is defined as $\eta_Y \circ f \colon X \to \mathcal{T}Y$. For the other

direction, the functor $U_{\mathcal{T}}$ maps an object X in $\mathsf{Kl}(\mathcal{T})$ to $\mathcal{T}X$ and a morphism $f\colon X \nrightarrow Y$ (which is seen here as a function $f\colon X \to \mathcal{T}Y$) to $\mu_Y \circ \mathcal{T}f\colon \mathcal{T}X \to \mathcal{T}Y$.

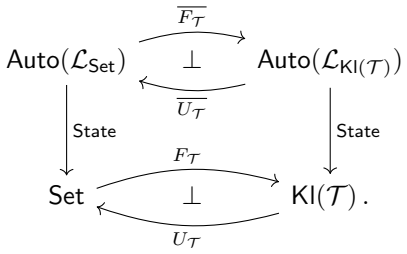
A simple, yet important observation is that the language of interest, which is a partial function $L\colon A^* \rightharpoonup B^*$ can be modeled either as a $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-language $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$, or, as a $(\mathsf{Set}, 1, B^* + 1)$-language $\mathcal{L}_{\mathsf{Set}}$. This is because for each $w \in A^*$ we can identify $L(w)$ either with an element of $\mathsf{Kl}(\mathcal{T})(1, 1)$ or, equivalently, as an element of $\mathsf{Set}(1, B^* + 1)$.

$$\begin{aligned} \mathcal{L}_{\mathsf{Kl}(\mathcal{T})}\colon\quad & \mathcal{O} \to \mathsf{Kl}(\mathcal{T}) \\ & \mathsf{in} \mapsto 1 \\ & \mathsf{out} \mapsto 1 \\ & \triangleright w \triangleleft \mapsto L(w)\colon 1 \nrightarrow 1 \end{aligned} \qquad\qquad \begin{aligned} \mathcal{L}_{\mathsf{Set}}\colon\quad & \mathcal{O} \to \mathsf{Set} \\ & \mathsf{in} \mapsto 1 \\ & \mathsf{out} \mapsto B^* + 1 \\ & \triangleright w \triangleleft \mapsto L(w)\colon 1 \to B^* + 1 \end{aligned}$$

To see how this fits in the scope of Section 3.1, notice that $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$ is a $(\mathsf{Kl}(\mathcal{T}), F_{\mathcal{T}}1, 1)$-language, while $\mathcal{L}_{\mathsf{Set}}$ is a $(\mathsf{Set}, 1, U_{\mathcal{T}}1)$-language and they correspond to each other via the bijections described in (1).

Applying Lemma 8 for the Kleisli adjunction (2) we obtain an adjunction $\overline{F_{\mathcal{T}}} \dashv \overline{U_{\mathcal{T}}}$ between the categories of $\mathsf{Kl}(\mathcal{T})$-automata for $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$ and of Set-automata accepting $\mathcal{L}_{\mathsf{Set}}$, as depicted in the picture below. The functor $\overline{U_{\mathcal{T}}}$ sends a $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automaton with state object Q to a $(\mathsf{Set}, 1, B^* + 1)$-automaton with state object $\mathcal{T}Q$, while $\overline{F_{\mathcal{T}}}$ sends a $(\mathsf{Set}, 1, B^* + 1)$-automaton with state object Q' to a $(\mathsf{Kl}(\mathcal{T}), 1, 1)$-automaton with same state object Q'.

We will make heavy use of this correspondence in what follows.



4.3 The initial $\mathsf{Kl}(\mathcal{T})$-automaton for the language $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$

The functor $\overline{F_{\mathcal{T}}}$ is a left adjoint and consequently preserves colimits and in particular the initial object. We thus obtain that the initial $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$-automaton is $\overline{F_{\mathcal{T}}}(\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}}))$, where $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})$ is the initial object of $\mathsf{Auto}(\mathcal{L}_{\mathsf{Set}})$. This automaton can be obtained by Lemma 7 as the functor $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})\colon \mathcal{I} \to \mathsf{Set}$ specified by $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})(\mathsf{states}) = A^*$ and for all $a \in A$

$$\begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})(\triangleright)\colon 1 &\to A^* \\ 0 &\mapsto \varepsilon \end{aligned} \qquad \begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})(\triangleleft)\colon A^* &\to B^* + 1 \\ w &\mapsto L(w) \end{aligned} \qquad \begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})(a)\colon A^* &\to A^* \\ w &\mapsto wa \end{aligned}$$

Hence, by computing the image of $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Set}})$ under $\overline{F_{\mathcal{T}}}$, we obtain the following description of the initial $\mathsf{Kl}(\mathcal{T})$-automaton $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ accepting $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$: $\mathcal{A}^{init}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\mathsf{states}) = A^*$ and for all $a \in A$

$$\begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\triangleright)\colon 1 &\nrightarrow A^* \\ 0 &\mapsto (\varepsilon, \varepsilon) \end{aligned} \qquad \begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\triangleleft)\colon A^* &\nrightarrow 1 \\ w &\mapsto L(w) \end{aligned} \qquad \begin{aligned} \mathcal{A}^{init}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(a)\colon A^* &\nrightarrow A^* \\ w &\mapsto (\varepsilon, wa) \end{aligned}$$

4.4 The final $\mathsf{Kl}(\mathcal{T})$-automaton for the language $\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}$

The case of the final $\mathsf{Kl}(\mathcal{T})$-automaton is more complicated, since it is not constructed as easily. However, assuming the final automaton exists, it has to be sent by $\overline{U_{\mathcal{T}}}$ to a final Set-automaton. Moreover, by Lemma 13, in order to prove that a given $\mathsf{Kl}(\mathcal{T})$-automaton $\mathcal{A}$ is a final object of $\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ it suffices to show that $\overline{U_{\mathcal{T}}}(\mathcal{A})$ is the final object in $\mathsf{Auto}(\mathcal{L}_{\mathsf{Set}})$. The proof of the following lemma generalises the fact that $U_{\mathcal{T}}$ reflects final objects and can be proved in the same spirit.

▶ **Lemma 13.** *The functor* $\overline{U_{\mathcal{T}}}\colon \mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}) \to \mathsf{Auto}(\mathcal{L}_{\mathsf{Set}})$ *reflects final objects.*

The final object in $\mathsf{Auto}(\mathcal{L}_{\mathsf{Set}})$ is the automaton $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})$ as described using Lemma 7. The functor $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})\colon \mathcal{I} \to \mathsf{Set}$ specified by

$$\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})(\mathsf{states}) = (B^*+1)^{A^*} \qquad \begin{aligned}\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})(\triangleleft)\colon (B^*+1)^{A^*} &\to B^*+1\\ K &\mapsto K(\varepsilon)\end{aligned}$$

$$\begin{aligned}\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})(\triangleright)\colon 1 &\to (B^*+1)^{A^*}\\ 0 &\mapsto L\end{aligned} \qquad \begin{aligned}\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})(a)\colon (B^*+1)^{A^*} &\to (B^*+1)^{A^*}\\ K &\mapsto \lambda w.K(aw)\end{aligned}$$

To describe the set of states of the final automaton in $\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ we need to introduce a few notations. Essentially we are looking for a set of states Q so that $B^* \times Q + 1$ is isomorphic to $(B^*+1)^{A^*}$. The intuitive idea is to decompose each function in $K \in (B^*+1)^{A^*}$ (except for the one which is nowhere defined, that is the function $\kappa_\bot = \lambda w.\bot$) into a word in B^*, the common prefix of all the B^*-words in the image of K, and an irreducible function.

For $v \in B^*$ and a function $K \neq \kappa_\bot$ in $(B^*+1)^{A^*}$, denote by $v \star K$ the function defined for all $u \in A^*$ by $(v \star K)(u) = v\,K(u)$ if $K(u) \in B^*$ and $(v \star K)(u) = \bot$ otherwise. Define also the longest common prefix of K, $\mathsf{lcp}(K) \in B^*$, as the longest word that is prefix of all $K(u) \neq \bot$ for u in A^* (this is well defined since $K \neq \kappa_\bot$). The reduction of K, $\mathsf{red}(K)$, is defined as:

$$\mathsf{red}(K)(u) = \begin{cases} v & \text{if } K(u) = \mathsf{lcp}(K)\,v,\\ \bot & \text{otherwise.}\end{cases}$$

Finally, K is called irreducible if $\mathsf{lcp}(K) = \varepsilon$ (or equivalently if $K = \mathsf{red}(K)$). We denote by $\mathsf{Irr}(A^*, B^*)$ the irreducible functions in $(B^*+1)^{A^*}$.

What we have constructed is a bijection between

$$\mathcal{T}(\mathsf{Irr}(A^*,B^*)) = B^* \times \mathsf{Irr}(A^*,B^*) + 1 \qquad \text{and} \qquad (B^*+1)^{A^*}\ ,$$

that is defined as

$$\begin{aligned}\varphi\colon B^* \times \mathsf{Irr}(A^*,B^*) + 1 &\to (B^*+1)^{A^*}\\ (u,K) &\mapsto u \star K\\ \bot &\mapsto \kappa_\bot\ ,\end{aligned} \tag{3}$$

and the converse of which maps every $K \neq \kappa_\bot$ to $(\mathsf{lcp}(K), \mathsf{red}(K))$, and $\kappa_\bot$ to $\bot$.

Given $a \in A$ and $K \in (B^*+1)^{A^*}$ we denote by $a^{-1}K$ the function in $(B^*+1)^{A^*}$ that maps $w \in A^*$ to $K(aw)$.

We can now define a functor $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})\colon \mathcal{I} \to \mathsf{Kl}(\mathcal{T})$ by setting

$$\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\mathsf{in}) = 1 \qquad \mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\mathsf{states}) = \mathsf{Irr}(A^*,B^*) \qquad \mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\mathsf{out}) = 1$$

and defining $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ on arrows as follows

$$\begin{aligned}
&\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\triangleright)\colon 1 \nrightarrow \mathsf{Irr}(A^*, B^*) && 0 \mapsto (\mathsf{lcp}(L), \mathsf{red}(L)) \\
&\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(\triangleleft)\colon \mathsf{Irr}(A^*, B^*) \nrightarrow 1 && K \mapsto K(\varepsilon) \\
&\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})(a)\colon \mathsf{Irr}(A^*, B^*) \nrightarrow \mathsf{Irr}(A^*, B^*) && K \mapsto \kappa_\perp \qquad \text{if } a^{-1}K = \kappa_\perp \\
& && K \mapsto (\mathsf{lcp}(a^{-1}K), \mathsf{red}(a^{-1}K)) \quad \text{otherwise.}
\end{aligned}$$

▶ **Lemma 14.** *The* $\mathsf{Kl}(\mathcal{T})$*-automaton* $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ *is a final object in* $\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$.

Proof. We show that $\overline{U_{\mathcal{T}}}(\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}))$ is isomorphic to the final automaton $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})$. Indeed, at the level of the state objects the bijection between $\overline{U_{\mathcal{T}}}(\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}))(\mathsf{states})$ and $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})(\mathsf{states})$ is given by the function φ defined in (3). One can check that on arrows $\overline{U_{\mathcal{T}}}(\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})}))$ is the same as $\mathcal{A}^{final}(\mathcal{L}_{\mathsf{Set}})$ up to the correspondence given by φ. ◀

4.5 A factorization system on $\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$

The factorization system on $\mathsf{Auto}(\mathcal{L}_{\mathsf{Kl}(\mathcal{T})})$ is obtained using Lemma 4 from a factorization system on $\mathsf{Kl}(\mathcal{T})$. There are several non-trivial factorization systems on $\mathsf{Kl}(\mathcal{T})$, one of which is obtained from the regular epi-mono factorization system on Set, or equivalently, from the regular epi-mono factorization system on the category of Eilenberg-Moore algebras for $\mathcal{T}$. Notice that this is a specific result for the monad $\mathcal{T}$ since in general, there is no reason that the Eilenberg-Moore algebra obtained by factorizing a morphism between free algebras be free itself. Nevertheless, in order to capture precisely the syntactic transducer defined by Choffrut [10, 11], we will provide yet another factorization system $(\mathcal{E}_{\mathsf{Kl}(\mathcal{T})}, \mathcal{M}_{\mathsf{Kl}(\mathcal{T})})$, which we define concretely as follows. Given a morphism $f\colon X \nrightarrow Y$ in $\mathsf{Kl}(\mathcal{T})$ we write $\pi_1(f)\colon X \to B^* + \{\perp\}$ and $\pi_2(f)\colon X \to Y + \{\perp\}$ for the 'projections' of f, defined by

$$\pi_1(f)(x) = \begin{cases} u & \text{if } f(x) = (u, y)\,, \\ \perp & \text{otherwise,} \end{cases} \qquad \text{and} \qquad \pi_2(f)(x) = \begin{cases} y & \text{if } f(x) = (u, y)\,, \\ \perp & \text{otherwise.} \end{cases}$$

We say that a partial function $g\colon X \to Y + \{\perp\}$ is surjective when for every $y \in Y$ there exists $x \in X$ so that $g(x) = y$.

The class $\mathcal{E}_{\mathsf{Kl}(\mathcal{T})}$ consists of all the morphisms of the form $e\colon X \nrightarrow Y$ such that $\pi_2(e)$ is surjective and the class $\mathcal{M}_{\mathsf{Kl}(\mathcal{T})}$ consists of all the morphisms of the form $m\colon X \nrightarrow Y$ such that $\pi_2(m)$ is injective and $\pi_1(m)$ is the constant function mapping every $x \in X$ to ε.

▶ **Lemma 15.** $(\mathcal{E}_{\mathsf{Kl}(\mathcal{T})}, \mathcal{M}_{\mathsf{Kl}(\mathcal{T})})$ *is a factorization system on* $\mathsf{Kl}(\mathcal{T})$.

Proof. Notice that f is an isomorphism in $\mathsf{Kl}(\mathcal{T})$ if and only if $f \in \mathcal{E}_{\mathsf{Kl}(\mathcal{T})} \cap \mathcal{M}_{\mathsf{Kl}(\mathcal{T})}$.

If $f\colon X \nrightarrow Y$ is a morphism in $\mathsf{Kl}(\mathcal{T})$ then we can define

$$Z = \{y \in Y \mid \exists x \in X.\, \exists u \in B^*.\, f(x) = (u, y)\}\,.$$

We define $e\colon X \nrightarrow Z$ by $e(x) = f(x)$ and $m\colon Z \nrightarrow Y$ by $m(y) = (\epsilon, y)$. One can easily check that $f = m \circ e$ in $\mathsf{Kl}(\mathcal{T})$.

Lastly, we can show that the "diagonal property" holds. Assume we have a commuting square in $\mathsf{Kl}(\mathcal{T})$.

$$\begin{array}{ccc} X & \xrightarrow{\;e\;} & Y \\ {\scriptstyle f}\downarrow & \swarrow_{d} & \downarrow{\scriptstyle g} \\ Z & \xrightarrow[\;m\;]{} & W \end{array}$$

We will prove the existence of $d\colon Y \nrightarrow Z$ so that $d \circ e = f$ and $m \circ d = g$. Assume $y \in Y$. If $g(y) = \bot$ we set $d(y) = \bot$. Otherwise assume $g(y) = (v, t)$, for some $v \in B^*$ and $t \in W$. Since $e \in \mathcal{E}_{\mathsf{Kl}(\mathcal{T})}$, there exists $u \in B^*$ and $x \in X$ so that $e(x) = (u, y)$. Assume $f(x) = (w, z)$ for some $w \in B^*$ and $z \in Z$. We set $d(y) = (v, z)$. First, we have to prove that this definition does not depend on the choice of x.

Assume that we have another $x' \in X$ so that $g(x') = (u', y)$ and assume $f(x') = (w', z')$. Using the fact that $m \in \mathcal{M}_{\mathsf{Kl}(\mathcal{T})}$, we will show that $z = z'$, and thus $d(y)$ is well defined. Indeed, notice that

$$\begin{cases} g \circ e(x) = (uv, t) \\ g \circ e(x') = (u'v, t)\,, \end{cases} \quad \text{or equivalently,} \quad \begin{cases} m \circ f(x) = (uv, t) \\ m \circ f(x') = (u'v, t)\,, \end{cases}$$

Assume that $m(z) = (\varepsilon, t_1)$ and $m(z') = (\varepsilon, t_2)$. This entails

$$\begin{cases} m \circ f(x) = (uv, t) = (w, t_1) \\ m \circ f(x') = (u'v, t) = (w', t_2)\,. \end{cases}$$

We obtain that $t_1 = t_2 = t$. Since $m \in \mathcal{M}_{\mathsf{Kl}(\mathcal{T})}$ (and thus $\pi_2(m)$ is injective) we get that $z = z'$, which is what we wanted to prove. It is easy to verify that $d \circ e = f$ and $m \circ d = g$. ◀

This completes the proof of Theorem 12.

5 Brzozowski's minimization algorithm

5.1 Presentation

Brzozowski's algorithm is a minimization algorithm for automata. It takes as input a non-deterministic automaton $\mathcal{A}$, and computes the deterministic automaton:

$$\texttt{determinize}(\texttt{transpose}(\texttt{determinize}(\texttt{transpose}(\mathcal{A})))),$$

in which

- `determinize` is the operation from classical automata theory that takes as input a deterministic automaton, performs the powerset construction to it and at the same time restricts to the reachable states, yielding a deterministic automaton, and
- `transpose` is the operation that takes as input a non-deterministic automaton, reverses all its edges, and swaps the role of initial and final states (it accepts the mirrored language).

In this section, we will establish the correctness of Brzozowski's algorithm: this sequence of operations yields the minimal automaton for the language. For easing the presentation we shall present the algorithm in the form:

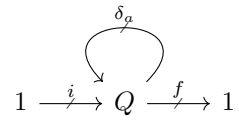
$$\texttt{determinize}(\texttt{codeterminize}(\mathcal{A})),$$

in which `codeterminize` is the operation that takes a non-deterministic automaton, and constructs a backward deterministic one (it is equivalent to the sequence `transpose` $\circ$ `determinize` $\circ$ `transpose`).

In the next section, we show how `determinize` and `codeterminize` can be seen as adjunctions, and we use it immediately after in a correctness proof of Brzozowski's algorithm.

5.2 Non-deterministic automata and determinization

A non-deterministic automaton is completely determined by the relations described in the following diagram, where we see the initial states as a relation from 1 to the set of states Q, the final states as a relation from Q to 1 and the transition relation by any input letter a, as a relation on Q.

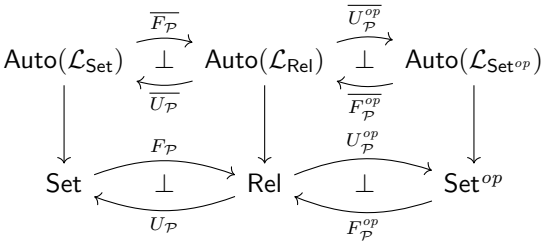


We can model nondeterministic automata as functors by taking as output category Rel – the category whose objects are sets and maps are relations between them. We consider Rel-automata $\mathcal{A}\colon \mathcal{I} \to \mathsf{Rel}$ such that $\mathcal{A}(\mathsf{in}) = 1$ and $\mathcal{A}(\mathsf{out}) = 1$. In this section we show how to determinize a Rel-automaton, that is, how to turn it into a Set-automaton and how to codeterminize it, that is, how to obtain a Set^{op}-automaton, all recognizing the same language.

Given a language $L \subseteq A^*$ we can model it in several equivalent ways: as a $(\mathsf{Set}, 1, 2)$-language $\mathcal{L}_{\mathsf{Set}}$, or as a $(\mathsf{Set}^{op}, 2, 1)$-language $\mathcal{L}_{\mathsf{Set}^{op}}$, or, lastly as a $(\mathsf{Rel}, 1, 1)$-language $\mathcal{L}_{\mathsf{Rel}}$. This is because we can model the fact $w \in L$ using a morphisms in either of the three isomorphic hom-sets

$$\mathsf{Set}(1,2) \cong \mathsf{Set}^{op}(2,1) \cong \mathsf{Rel}(1,1)\,.$$

Determinization and codeterminization (without assuming the restriction to reachable states as in `determinize` and `codeterminize`) of a Rel-automaton can be seen as applications of Lemma 8 and are obtained by lifting the adjunctions between Set, Rel and Set^{op}.



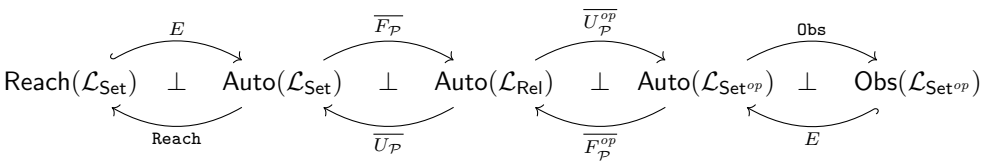
(4)

The adjunction between Set and Rel is the Kleisli adjunction for the powerset monad: $F_{\mathcal{P}}$ is identity on objects and maps a function $f\colon X \to Y$ to itself $f\colon X \nrightarrow Y$, but seen as a relation. The functor $U_{\mathcal{P}}$ maps X to its powerset $\mathcal{P}(X)$, and a relation $R\colon X \to Y$ to the function $U_{\mathcal{P}}(R)\colon \mathcal{P}(X) \to \mathcal{P}(Y)$ mapping $A \subseteq X$ to $\{y \in Y \mid \exists x \in A.(x,y) \in R\}$.

The adjunction between Set^{op} and Rel is the dual of the previous one, composed with the self-duality of Rel. The left adjoint $\overline{F_{\mathcal{P}}}$ transforms a deterministic automaton into a non-deterministic one, while the right adjoint $\overline{U_{\mathcal{P}}}$ is the determinization functor. On the other hand, the left adjoint functor $\overline{U^{op}_{\mathcal{P}}}$ is the codeterminization functor.

5.3 Brzozowski's minimization algorithm

The correctness of Brzozowski's algorithm can be seen in the following chain of adjunctions from Lemma 3 and (4) (that all correspond to equivalences at the level of languages):



A path in this diagram corresponds to a sequence of transformations of automata. It happens that when `Obs` is taken, the resulting automaton is observable, i.e., there is an injection from it to the final object. This property is preserved under the sequence of right adjoints $\texttt{Reach} \circ \overline{U_{\mathcal{P}}} \circ \overline{F_{\mathcal{P}}^{op}} \circ E$. Furthermore, after application of `Reach`, the automaton is also reachable. This means that applying the sequence $\texttt{Reach} \circ \overline{U_{\mathcal{P}}} \circ \overline{F_{\mathcal{P}}^{op}} \circ E \circ \texttt{Obs} \circ \overline{U_{\mathcal{P}}^{op}}$ to a non-deterministic automaton produces a deterministic and minimal one for the same language. We check for concluding that the sequence $\texttt{Obs} \circ \overline{U_{\mathcal{P}}^{op}}$ is what is implemented by `codeterminize`, that the composite $\overline{F_{\mathcal{P}}^{op}} \circ E$ essentially transforms a backward deterministic observable automaton into a non-deterministic one, and that finally $\texttt{Reach} \circ \overline{U_{\mathcal{P}}}$ is what is implemented by `determinize`. Hence, this indeed is Brzozowski's algorithm.

▶ Remark. The composite of the two adjunctions in (4) is almost the adjunction of [7, Corollary 9.2] upon noticing that the category $\mathsf{Auto}(\mathcal{L}_{\mathsf{Set}^{op}})$ of Set^{op}-automata accepting a language $\mathcal{L}_{\mathsf{Set}^{op}}$ is isomorphic to the opposite of the category $\mathsf{Auto}(\mathcal{L}^{\mathsf{rev}}_{\mathsf{Set}})$ of Set-automata that accept the reversed language seen as functor $\mathcal{L}_{\mathsf{Set}^{op}}$. This observation in turn can be proved using the symmetry of the input category $\mathcal{I}$.

6 Conclusion

In this paper we propose a view of automata as functors and we showed how to recast well understood classical constructions in this setting, and in particular minimization of subsequential transducers. The applications provided here are a small sample of many possible further extensions. We argue that this perspective gives a unified view of language recognition and syntactic objects. We can change the input category $\mathcal{I}$, so that we obtain monoids instead of automata, or more generally, other algebras as recognisers for languages. Minimization works out following the same recipe and yields the syntactic monoid (algebra) of a language. We can go beyond regular languages and obtain in this fashion the "syntactic space with an internal monoid" of a possibly non-regular language [13]. Our functorial treatment of automata is more general than that presented in [5] and it would be interesting to explore equations and coequations in this setting. We hope we can extend the framework to work with tree automata in monoidal categories. We discussed mostly NFA determinization, but we can obtain a variation of the generalized powerset construction [19] in this framework.

References

1 Jiří Adámek and Věra Trnková. *Automata and Algebras in Categories*. 37. Springer Netherlands, New York, 1989. URL: `http://www.springer.com/fr/book/9780792300106`.

2 Jiří Adámek, Filippo Bonchi, Mathias Hülsbusch, Barbara König, Stefan Milius, and Alexandra Silva. A coalgebraic perspective on minimization and determinization. In *Proceedings of the 15th International Conference on Foundations of Software Science and Computational Structures*, FOSSACS'12, pages 58–73, Berlin, Heidelberg, 2012. Springer-Verlag.

3 Michael A. Arbib and Ernest G. Manes. Adjoint machines, state-behavior machines, and duality. *Journal of Pure and Applied Algebra*, 6(3):313 – 344, 1975. `doi:10.1016/0022-4049(75)90028-6`.

4 E. S. Bainbridge. Adressed machines and duality. In *Category Theory Applied to Computation and Control*, volume 25 of *Lecture Notes in Computer Science*, pages 93–98. Springer, 1974.

5 Adolfo Ballester-Bolinches, Enric Cosme-Llópez, and Jan J. M. M. Rutten. The dual equivalence of equations and coequations for automata. *Inf. Comput.*, 244:49–75, 2015.

6 Nick Bezhanishvili, Clemens Kupke, and Prakash Panangaden. *Minimization via Duality*, pages 191–205. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi:10.1007/978-3-642-32621-9_14.

7 Filippo Bonchi, Marcello M. Bonsangue, Helle Hvid Hansen, Prakash Panangaden, Jan J. M. M. Rutten, and Alexandra Silva. Algebra-coalgebra duality in Brzozowski's minimization algorithm. *ACM Trans. Comput. Log.*, 15(1):3:1–3:29, 2014.

8 Filippo Bonchi, Marcello M. Bonsangue, Jan J. M. M. Rutten, and Alexandra Silva. Brzozowski's algorithm (co)algebraically. In *Logic and Program Semantics*, volume 7230 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 2012.

9 C. Cassidy, M. Hébert, and G. M. Kelly. Reflective subcategories, localizations and factorizationa systems. *Journal of the Australian Mathematical Society. Series A. Pure Mathematics and Statistics*, 38(3):287–329, 1985. doi:10.1017/S1446788700023624.

10 Christian Choffrut. A generalization of Ginsburg and Rose's characterization of G-S-M mappings. In *ICALP*, volume 71 of *Lecture Notes in Computer Science*, pages 88–103. Springer, 1979.

11 Christian Choffrut. Minimizing subsequential transducers: a survey. *Theoretical Computer Science*, 292(1):131 – 143, 2003. doi:10.1016/S0304-3975(01)00219-5.

12 Thomas Colcombet and Daniela Petrişan. Automata in glued vector spaces. submitted, 2017.

13 Mai Gehrke, Daniela Petrisan, and Luca Reggio. The Schützenberger product for syntactic spaces. In *ICALP*, volume 55 of *LIPIcs*, pages 112:1–112:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

14 J. A. Goguen. Minimal realization of machines in closed categories. *Bull. Amer. Math. Soc.*, 78(5):777–783, 09 1972. URL: http://projecteuclid.org/euclid.bams/1183533991.

15 Helle Hvid Hansen. Subsequential transducers: a coalgebraic perspective. *Inf. Comput.*, 208(12):1368–1397, 2010.

16 Bart Jacobs and Jan Rutten. A tutorial on (co)algebras and (co)induction. *EATCS Bulletin*, 62:62–222, 1997.

17 Bart Jacobs, Alexandra Silva, and Ana Sokolova. Trace semantics via determinization. *Journal of Computer and System Sciences*, 81(5):859 – 879, 2015. 11th International Workshop on Coalgebraic Methods in Computer Science, {CMCS} 2012 (Selected Papers). doi:10.1016/j.jcss.2014.12.005.

18 Henning Kerstan, Barbara König, and Bram Westerbaan. Lifting adjunctions to coalgebras to (re)discover automata constructions. In *CMCS*, volume 8446 of *Lecture Notes in Computer Science*, pages 168–188. Springer, 2014.

19 Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Logical Methods in Computer Science*, 9(1), 2013.

The Positivication of Coalgebraic Logics

Fredrik Dahlqvist[1] **and Alexander Kurz**[2]

1 **University College London, UK**
`f.dahlqvist@ucl.ac.uk`
2 **University of Leicester, UK**
`ak155@leicester.ac.uk`

Abstract

We present positive coalgebraic logic in full generality, and show how to obtain a positive coalgebraic logic from a boolean one. On the model side this involves canonically computing a endofunctor $T' : \mathbf{Pos} \to \mathbf{Pos}$ from an endofunctor $T : \mathbf{Set} \to \mathbf{Set}$, in a procedure previously defined by the second author *et alii* called *posetification*. On the syntax side, it involves canonically computing a syntax-building functor $L' : \mathbf{DL} \to \mathbf{DL}$ from a syntax-building functor $L : \mathbf{BA} \to \mathbf{BA}$, in a dual procedure which we call *positivication*. These operations are interesting in their own right and we explicitly compute posetifications and positivications in the case of several modal logics. We show how the semantics of a boolean coalgebraic logic can be canonically lifted to define a semantics for its positive fragment, and that weak completeness transfers from the boolean case to the positive case.

1998 ACM Subject Classification I.2.4 Knowledge Representation Formalisms and Methods – Modal logic, temporal logic

Keywords and phrases Coalgebraic logic, coalgebras, enriched category theory, boolean algebra, distributive lattice, positive modal logic, monotone modal logic

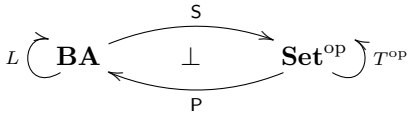
Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.9

1 Introduction

Partially ordered structures are ubiquitous in theoretical computer science. From knowledge representation to abstract interpretation in static analysis, from resource modelling to protocol or access rights formalization in formal security, the list of applications is enormous. Being able to formally reason about transition systems over posets therefore seems important, but has not been systematically developed. The natural formalism to reason about transition systems is undoubtedly the class of *modal* logics. However, most are tailored to transition structures over *sets*. This is a direct consequence of the fact that most modal logics are *boolean*. *Positive modal logic* is the exception, and is most naturally interpreted in partially ordered Kripke structures (see for example [6, 9]).

Arguably, the most natural and powerful framework to study boolean modal logics in a uniform and systematic way, is the theory of *Boolean Coalgebraic Logics* (henceforth BCL, see e.g. [7]). In its 'abstract' ([19]) presentation, it is parametrised by an endofunctor $L : \mathbf{BA} \to \mathbf{BA}$ which builds modal algebras of modal terms over a boolean structure, an endofunctor $T : \mathbf{Set} \to \mathbf{Set}$ which builds the transition structures over which the modal terms are to be interpreted, and a natural transformation $\delta : L\mathsf{P} \to \mathsf{P}T^{\mathrm{op}}$ (where $\mathsf{P} : \mathbf{Set}^{\mathrm{op}} \to \mathbf{BA}$ is the powerset functor) which implements the interpretation by associating sets of acceptable

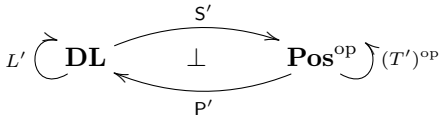
successors states to each modal term over a predicate (see [17, 18, 12, 21]). This data, and the dual adjunction between **Set** and **BA**, is traditionally summarized in the following diagram



$$\delta : L\mathsf{P} \Rightarrow \mathsf{P}T^{\mathrm{op}} \tag{1}$$

where S is the functor sending a boolean algebra to the set of its ultrafilters.

To develop an equally powerful framework for reasoning about transition structures over posets, it seems natural to study *Positive Coalgebraic Logics* (henceforth PCL). In fact, work in this direction has already started, see for example [14, 3]. We pursue this work further and present PCL in full generality, i.e. at the same level of generality as its boolean counterpart. Moreover, given the close kinship between the two theories, we will show that the wheel needn't be re-invented every time, and that many BCLs have a canonical positive fragment which inherits useful properties from its boolean parent. The data defining a PCL will be



$$\delta' : L'\mathsf{P}' \Rightarrow \mathsf{P}'(T')^{\mathrm{op}} \tag{2}$$

where S' is the functor sending a distributive lattice to the poset of its prime filters, and P' is the functor sending a poset to the distributive lattice of its upsets. The following observation will be of fundamental importance in what follows: the adjunction $\mathsf{S}' \dashv \mathsf{P}' : \mathbf{Pos}^{\mathrm{op}} \to \mathbf{DL}$ which is the backbone of diagram (2) is in fact **Pos**-*enriched*; that is to say **DL** and **Pos** are **Pos**-enriched categories and S', P' are **Pos**-enriched functors ([16]). Clearly, it would be a shame not to use this extra structure which comes for free. But more seriously, this enriched structure is not simply a mathematical quirk, it suggests that 'doing logic' positively is quite different from 'doing logic' in a boolean setting, in particular it is more than simply dropping negations. In fact *inequations* become the standard relation between terms on the syntax side, just as it is between elements on the model side. This is borne out by the existing axiomatization of positive modal logic originally proposed in [8] which is entirely given by inequations. For these reasons, and following [14, 3], this paper will present positive coalgebraic logic as a **Pos**-enriched coalgebraic logic. In a slogan: "*in positive coalgebraic logic we remove negations but we add order*".

Working in a **Pos**-enriched setting means that the syntax-building functor $L' : \mathbf{DL} \to \mathbf{DL}$ and the coalgebra-building functor $T' : \mathbf{Pos} \to \mathbf{Pos}$ will also need to be **Pos**-enriched. The first main contributions of this paper is to show how well-known ordinary functors from BCL can be turned into **Pos**-enriched functors performing analogous roles in PCL. On the semantics side this means turning an ordinary functor $T : \mathbf{Set} \to \mathbf{Set}$ into a **Pos**-enriched functor $T' : \mathbf{Pos} \to \mathbf{Pos}$ by a process called *posetification* first developed in [3], for which we develop a practical understanding in Section 3 by computing the posetification of several well-known functors from modal logic: the neighbourhood, monotone neighbourhood, powerset and multiset functors. On the syntax side this means turning an ordinary functor $L : \mathbf{BA} \to \mathbf{BA}$ into a **Pos**-enriched functor $L' : \mathbf{DL} \to \mathbf{DL}$, a process which we call *positivication* and which is detailed in Section 4. We show how positivication can be applied to the functor defining normal modal logic, but also to functors which define non-monotone modal logics. In this case, the positivication procedure may not yield a logic at all, at least not in the usual meaning of the word. The second main contribution of this paper is to show how a semantic natural transformation $\delta : L\mathsf{P} \to \mathsf{P}T$ can also be lifted to define a

Pos-enriched semantic natural transformation $\delta' : L'\mathsf{P}' \to \mathsf{P}'T'$, where T' is the posetification of T, and L' is the positivication of L. This is done in Section 5, where we also prove that by construction of δ', if δ defines a weakly complete logic, then so does δ'.

Related work. As mentioned above several aspects of positive coalgebraic logics have been studied in [14, 3], the concept of posetification and the idea of working in an enriched setting in particular. The key contribution of this work is to dissociate the syntax from the semantics. This reflects the practise of modal logics, where the syntax-building functor L is usually not defined directly from the semantics-building functor, but rather from a grammar which is convenient to express certain properties. Graded modal logic for example relies on a syntax which is not obviously related to its semantics. This justifies going beyond the techniques of [3]. As a consequence, one must also be able to define a semantic natural transformation $L'\mathsf{P}' \to \mathsf{P}'T'$, which we do be adapting the boolean semantics. We are also indebted to work on monotone modal logic for the monotone neighbourhood functor, see eg [11, 23], and on non-monotone modal logic for the (unrestricted) neighbourhood functor as these two cases highlight many of the peculiar features of our approach.

2 A maths toolkit

2.1 Ordinary vs Pos-enriched category theory

The central tool of this paper is to work in categories enriched over **Pos**. For a general reference to enriched categories we refer to [16]. But the special case of **Pos**-categories is much simpler than the general case and we believe that most of this paper can be read without special knowledge in enriched category theory. The purpose of this section is to review what will be required.

A **Pos**-category is a category in which homsets are posets and composition is monotone in each argument. A **Pos**-functor is a functor that is *locally monotone*, that is, it preserves the order on homsets. **Pos**-natural transformations are just natural transformations.

Monotonicity permeates all aspects of **Pos**-enriched categories. For example, **Pos**-enriched algebra, or *ordered algebra*, is characterised by all operations of an ordered algebra being monotone. This is important for our application of ordered algebra to positive coalgebraic logic, that is, to coalgebraic logic with monotone operations.

The basic features of ordered universal algebra can be developed in much the same way as ordinary universal algebra [5]. Following the **Pos**-enriched approach of [22], the most important change to make is to replace coequalisers by so-called coinserters.

One of the most important features of the **Pos**-enriched setting is that with the so-called weighted limits and colimits additional universal constructions become available. For example,

$$A \underset{f}{\overset{g}{\underset{\uparrow}{\rightrightarrows}}} B \xrightarrow{c} C$$

is a *coinserter* if $c \circ f \leq c \circ g$ and for all h with $h \circ f \leq h \circ g$ there is a unique k such that $k \circ c = h$. This is almost like a coequaliser, but C is a quotient of B w.r.t. inequations. For example, in **Pos**, the coinserter C is obtained by adding to B the inequations $\{fa \leq ga \mid a \in A\}$ and then quotienting by anti-symmetry. We will encounter two special kinds of coinserters (we sometimes drop now the $\uparrow$ notation in the interest of typesetting):

$$s \circlearrowleft A \underset{\pi_0}{\overset{\pi_1}{\rightrightarrows}} B \qquad A \xrightarrow{i} B$$

A pair of arrows, and also its coinserter, is called *reflexive* if there is $i : B \to A$ such that $\pi_0 \circ i = \pi_1 \circ i = \mathrm{id}$ and it is called *symmetric* if there is $s : A \to A$ such that $\pi_0 \circ s = \pi_1$ and $\pi_1 \circ s = \pi_0$. Note that an arrow is a coinserter of a symmetric pair, or a symmetric coinserter for short, iff it is a coequaliser of the same pair.

The dual notion is that of an *inserter*

$$E \xrightarrow{\ e\ } B \underset{f}{\overset{g}{\uparrow}} A.$$

In **Pos**, as well as in categories of ordered algebras, the inserter E is $\{b \in B \mid fa \leq gb\}$.

Another important limit not available in ordinary categories is the *power* or *cotensor* with a poset. For example, in **Pos** we have $X^{\mathbb{2}} = \{(x, x') \mid x \leq x'\}$ where $\mathbb{2}$ is $\{0 < 1\}$. We will also encounter the dual notion, the *tensor* or *copower* $X \bullet A$. Here we say that a category $\mathcal{A}$ has tensors if for all $A \in \mathcal{A}$ and all posets X, there is an object $X \bullet A$ such that

$$\mathcal{A}(X \bullet A, A') \cong [X, \mathcal{A}(A, A')]$$

where $[X, Y]$ denotes exponentiation (aka internal hom) in **Pos**. The tensor $\mathbb{2} \bullet A$ can be understood as an ordered coproduct of A with itself in which "each a on the left is smaller than the a on the right".

In order to treat ordinary categories and **Pos**-enriched categories in the same framework, we consider an ordinary category as a **Pos**-category with discrete homsets. For each **Pos**-category $\mathcal{A}$ there is a corresponding ordinary category $\mathcal{A}_o$. For example, we have $\mathbf{Set} = \mathbf{Set}_o$ and $\mathbf{BA} = \mathbf{BA}_o$, but **Pos** and $\mathbf{Pos}_o$ are different. In particular, there is no (enriched) forgetful functor $\mathbf{Pos} \to \mathbf{Set}$, only an (ordinary) forgetful functor $\mathbf{Pos}_o \to \mathbf{Set}_o$. Note that using "$_o$" allows us to drop the qualifications enriched and ordinary without creating ambiguity. For example, the inclusion $\mathsf{D} : \mathbf{Set} \to \mathbf{Pos}$ has a left adjoint $\mathsf{C} : \mathbf{Pos} \to \mathbf{Set}$ mapping a poset to its connected components and the inclusion $\mathsf{D}_o : \mathbf{Set}_o \to \mathbf{Pos}_o$ has as a right adjoint the forgetful functor $\mathsf{V} : \mathbf{Pos}_o \to \mathbf{Set}_o$, but $\mathsf{D} : \mathbf{Set} \to \mathbf{Pos}$ does not have a right adjoint.[1]

$$\mathsf{C} \dashv \mathsf{D} : \mathbf{Set} \to \mathbf{Pos} \qquad \mathsf{D}_o \dashv \mathsf{V} : \mathbf{Pos}_o \to \mathbf{Set}_o \tag{3}$$

Accordingly, D preserves all (weighted) limits and D_o preserves all (ordinary) colimits. But D does not preserve all (**Pos**-enriched) colimits and indeed we will see later that D does not preserve all coinserters.

We will also need the corresponding results on the algebraic side. The inclusion $\mathsf{W} : \mathbf{BA} \to \mathbf{DL}$ has a right adjoint K (mapping a **DL** to the largest Boolean subalgebra it contains) and $\mathsf{W}_o : \mathbf{BA}_o \to \mathbf{DL}_o$ has a left-adjoint $G : \mathbf{DL}_o \to \mathbf{BA}_o$ (mapping a distributive lattice to the free **BA** over it).

$$\mathsf{K} \vdash \mathsf{W} : \mathbf{BA} \to \mathbf{DL} \qquad \mathsf{W}_o \vdash \mathsf{G} : \mathbf{DL}_o \to \mathbf{BA}_o \tag{4}$$

Note that (3) and (4) are dually equivalent when restricted to finite structures.

2.2 The ordered variety of Boolean algebras

The category **BA** of Boolean algebras has discrete homsets, giving rise to a forgetful functor $\mathbf{BA} \to \mathbf{Set}$. This functor is a **Pos**-enriched, or ordered, variety [22]. At the heart of this

[1] $L : \mathcal{A} \to \mathcal{B}$ is a **Pos**-enriched left-adjoint of $R : \mathcal{B} \to \mathcal{A}$ if there is a natural isomorphism *of posets* $\mathcal{B}(LA, B) \cong \mathcal{A}(A, RB)$. We have $\mathbf{Pos}_o(DA, B) \cong \mathbf{Set}_o(A, \mathsf{V}B)$ but not $\mathbf{Pos}(DA, B) \cong \mathbf{Set}(A, \mathsf{V}B)$.

observation is the fact that reflexive coinserters in **BA** are symmetric, a fact that we prove in detail as it will be important later.

▶ **Proposition 1.** *Every reflexive pair* $A_1 \underset{\pi_0}{\overset{\pi_1}{\rightrightarrows}} A_0$ (i : $A_0 \to A_1$) *in* **BA** *is symmetric.*

Proof. Let $(a,b) \in A_1$. Here, (a,b) is a shorthand for an element in A_1 such that $\pi_0((a,b)) = a$ and $\pi_1((a,b)) = b$. We write (a,a) for $i(a)$. Consider the Boolean algebra morphism $\phi : A_1 \to A_1$ defined by

$$\phi(a,b) = \Big((a,b) \wedge (b,b) \to (a,a)\Big) \wedge \Big((a,b) \wedge (a,a) \to (b,b)\Big) \wedge \Big((a,a) \vee (a,b) \vee (b,b)\Big)$$

Since the projections are **BA**-morphisms, we obtain $\pi_0(\phi(a,b)) = (a \to b) \wedge (a \vee b) = b$ and $\pi_1(\phi(a,b)) = (b \to a) \wedge (a \vee b) = a$, showing $(b,a) \in A_1$. ◀

(The argument also works for Heyting algebras.) Note that we can equip **BA** with a forgetful functor $\mathbf{BA} \xrightarrow{\mathsf{W}} \mathbf{DL} \to \mathbf{Pos}$ mapping each BA to its carrier in its natural order, but this functor is not an ordered variety since **BA** is not closed under weighted limits in **DL**.

2.3 Density and Kan extensions

Much of our technical work revolves around the result that $\mathsf{D} : \mathbf{Set} \to \mathbf{Pos}$ is dense, see [3], and that $\mathsf{W} : \mathbf{BA} \to \mathbf{DL}$ is codense, see Theorem 12. This in turn allows us to extend functors on **Set** to functors on **Pos** via left Kan extension and to extend functors on **BA** to functors on **DL** via right Kan extensions, as we will review now.

A functor $K : \mathcal{A} \to \mathcal{C}$ is *dense* if colimit preserving functors $\mathcal{C} \to \mathcal{B}$ are determined by their restriction along K, or, more formally, if the functor $[K, \mathsf{Id}_{\mathcal{B}}] : [\mathcal{C}, \mathcal{B}] \to [\mathcal{A}, \mathcal{B}]$ restricting along K is fully faithful [16, Thm 5.1]. If, moreover, K itself is fully faithful, then a colimit preserving functor $\mathcal{C} \to \mathcal{B}$ is the left Kan extension of its restriction along K [16, Thm 5.29]. Furthermore, we may be able to compute left Kan extensions explicitely with the help of a so-called density presentation [16, Thm 5.19]. For example, we know (see [3]) that $\mathsf{D} : \mathbf{Set} \to \mathbf{Pos}$ is dense and has a density presentation given by reflexive coinserters of 'nerves of posets'. Explicitly, every poset X is the reflexive coinserter[2]

$$\mathsf{D}\mathsf{V}X^{\mathbb{2}} \underset{\mathsf{D}\pi_0}{\overset{\mathsf{D}\pi_1}{\rightrightarrows}} \mathsf{D}\mathsf{V}X \longrightarrow X \qquad (i : \mathsf{D}\mathsf{V}X \to \mathsf{D}\mathsf{V}X^{\mathbb{2}}) \tag{5}$$

where $\mathsf{V}X^{\mathbb{2}} = \{(x,x') \mid x \leq_X x'\}$. That the coinserter is reflexive means that $\mathsf{D}\pi_0 \circ i = \mathsf{D}\pi_1 \circ i = \mathrm{id}$, which is true for $i(x) = (x,x)$. The fact that these coinserters provide a density presentation means that the left Kan extension of a functor $F : \mathbf{Set} \to \mathbf{Pos}$ along D can be computed as the coinserter

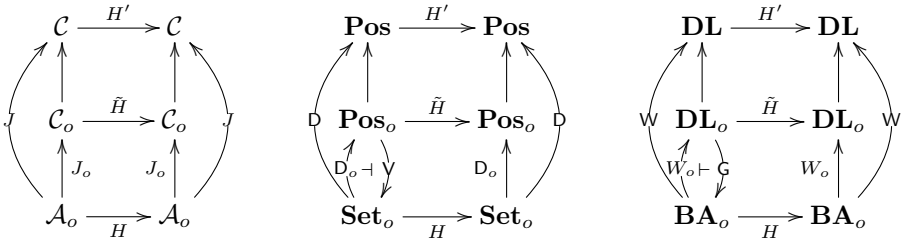
$$F\mathsf{V}X^{\mathbb{2}} \underset{F\pi_0}{\overset{F\pi_1}{\rightrightarrows}} F\mathsf{V}X \longrightarrow (\mathrm{Lan}_{\mathsf{D}} F)X \tag{6}$$

and we will see examples of this in the next section. If one happens to extend along an adjoint functor K, Kan-extensions are easier:

$$K \dashv V \implies \mathrm{Lan}_K F = FV \qquad G \dashv K \implies \mathrm{Ran}_K F = FG \tag{7}$$

[2] The coinserter of $VX^{\mathbb{2}} \rightrightarrows VX$ in **Set** provides an example of weighted colimit that is not preserved by D, showing that D cannot have an (enriched) right adjoint.

This implies that to compute (ordinary) Kan extensions along D_o or W_o we can use (3) or (4), and (7). To better understand the difference with extending along D or W, we can see the computation of the Kan extensions in two steps:



(8)

(i) Since $\mathcal{C}_o \to \mathcal{C}$ preserves all ordinary (co)limits we can use [16, Thm 4.47] to break down the extension of JH along J into first extending J_oH along J_o to $\tilde{H}$ and then extending $\tilde{H}$ to H'. (ii) If a functor $\mathcal{C}_o \to \mathcal{C}_o$ is locally monotone then this functor is its own extension (both left and right) to $\mathcal{C} \to \mathcal{C}$. This means that for locally monotone functors $\tilde{H}$ the upper square is trivial. (iii) To compute $\tilde{H}$ we can use (3) or (4), and (7). — While this is sometimes a good approach, the downside is that $\tilde{H}$ is typically not locally monotone (so we cannot use (i)) and the inclusion $\mathcal{C}_o \to \mathcal{C}$ is not fully faithful (so we loose the good properties of Kan extensions along fully faithful functors). To summarise, to compute Kan extensions along D we use (6) and for W we will develop a similar presentation in Theorem 12.

3 Posetification

We define the *posetification* T' *of a* **Set***-functor* T as $\mathrm{Lan}_{\mathsf{D}}\mathsf{D}T$, the left Kan-extension of $\mathsf{D}T$ along D, together with a natural isomorphism $\alpha : \mathsf{D}T \Rightarrow T'\mathsf{D}$. In concrete examples we will typically define T' so that α is the identity. In particular, T' is the universal locally monotone extension of T, that is, for all $S : \mathbf{Pos} \to \mathbf{Pos}$ and $\beta : \mathsf{D}T \Rightarrow S\mathsf{D}$, there is a unique $\gamma : T' \to S$ such that $\gamma \circ \alpha = \beta$. The coinserter (6) now becomes

$$\mathsf{D}T\mathsf{V}X^{\mathbb{2}} \underset{\mathsf{D}T\pi_0}{\overset{\mathsf{D}T\pi_1}{\rightrightarrows}} \mathsf{D}T\mathsf{V}X \xrightarrow{e_X} T'X. \tag{9}$$

It is computed for any poset X in the following way:

(i) consider the (reflexive) relation $R_T \subseteq T\mathsf{V}X \times T\mathsf{V}X$ given by

$$(a,b) \in R_T \Leftrightarrow \exists c \in T\mathsf{V}X^{\mathbb{2}}.\ T\pi_0(c) = a \ \&\ T\pi_1(c) = b$$

(ii) compute its transitive closure $\leq_T$

(iii) quotient $T\mathsf{V}X$ by the equivalence relation $\equiv_T = \leq_T \cap \geq_T$,

(iv) the coinserter is given by $(T\mathsf{V}X/\equiv_T, \leq_T)$.

Note that by definition, $\mathsf{V}X^{\mathbb{2}} \subseteq \mathsf{V}X \times \mathsf{V}X$ is precisely the graph of the partial order on X. It follows that R_T is simply the *lifting of the partial order on* X *by the functor* T (see [3, Remark 4.8]), often denoted $\overline{T}{\leq}$. In the rest of this section we will see examples where R_T is transitive, where R_T is transitive and antisymmetric, and where R_T is not even transitive.

3.1 Posetification of the covariant powerset functor $\mathcal{P}$

We recall this case here from [3] because it illustrates the steps (ii)-(iv) of the posetification procedure very clearly. We start by defining the relation $R_{\mathcal{P}} \subseteq \mathcal{P}\mathsf{V}X \times \mathcal{P}\mathsf{V}X$ by

$$(a,b) \in R_{\mathcal{P}} \Leftrightarrow \exists c \in \mathcal{P}\mathsf{V}X^{\mathbb{2}}.\ \pi_0[c] = a \ \&\ \pi_1[c] = b$$

which means that

$$(a,b) \in R_{\mathcal{P}} \iff (\forall x \in a)(\exists y \in b)\,.\, x \leq y \text{ and } (\forall y \in b)(\exists x \in a)\,.\, x \leq y$$

It is well-known that $\mathcal{P}$ preserves weak-pullbacks, and that this guarantees the transitivity of $R_{\mathcal{P}}$. We can thus skip step (i) of the posetification procedure since $R_{\mathcal{P}} = \leq_{\mathcal{P}}$. The relation $R_{\mathcal{P}} = \overline{\mathcal{P}}\leq$ is known as the Egli-Milner (pre-)order associated with $\leq$. It is not hard to check that $a \equiv_{\mathcal{P}} b$, i.e. $a \leq_{\mathcal{P}} b$ and $b \leq_{\mathcal{P}} a$, iff $\mathrm{Conv}(a) = \mathrm{Conv}(b)$ where $\mathrm{Conv}(a)$ is the convex closure of a, i.e. the set $\{x \in X \mid \exists y_1, y_2 \in a, y_1 \leq x \leq y_2\}$, and that $a \equiv_{\mathcal{P}} \mathrm{Conv}(a)$. It follows that $\mathcal{P}'X = (\mathcal{P}\mathsf{V}X/\equiv_{\mathcal{P}}, R_{\mathcal{P}}) = (\{\mathrm{Conv}(a) \mid a \subseteq X\}, R_{\mathcal{P}})$.

3.2 Posetification of analytic functors

Consider first a polynomial functor $\mathsf{H}_\Sigma : \mathbf{Set} \to \mathbf{Set}$ given by a signature $\mathrm{ar} : \Sigma \to \mathbb{N}$ and a collection of set $(A_\sigma)_{\sigma\in\Sigma}$

$$\begin{cases} \mathsf{H}_\Sigma X = \coprod_{\sigma\in\Sigma} A_\sigma \times X^{\mathrm{ar}(\sigma)} \\ \mathsf{H}_\Sigma f = \coprod_{\sigma\in\Sigma} \mathrm{id}_{A_\sigma} \times f^{\mathrm{ar}(\sigma)} \end{cases}$$

By definition of H_Σ on morphisms, the relation $R_{\mathsf{H}} \subseteq \mathsf{H}_\Sigma\mathsf{V}X \times \mathsf{H}_\Sigma\mathsf{V}X$ is given by

$$\big((a, x_1, \ldots, x_{\mathrm{ar}(\sigma)}), (b, y_1, \ldots, y_{\mathrm{ar}(\sigma')})\big) \in R_{\mathsf{H}} \Leftrightarrow a = b, \sigma = \sigma', x_i \leq y_i, 1 \leq i \leq \mathrm{ar}(\sigma)$$

Since polynomial functors preserve weak-pullbacks we have $R_{\mathsf{H}} = \overline{\mathsf{H}_\Sigma}\leq = \leq_{\mathsf{H}}$. Moreover, it is easy to see from the definition above that $\leq_{\mathsf{H}}$ is anti-symmetric (since $\leq$ is). It follows that $\mathsf{H}'_\Sigma X$ is simply given by $(\mathsf{H}_\Sigma\mathsf{V}X, R_{\mathsf{H}})$.

We can now compute the posetification of *analytic functors* ([13]), i.e. functors of the shape

$$\mathsf{G}_\Sigma X = \coprod_{\sigma\in\Sigma} A_\sigma \times (X^{\mathrm{ar}(\sigma)}/G_\sigma)$$

where each quotient $X^{\mathrm{ar}(\sigma)}/G_\sigma$ is taken with respect to the obvious action of a subgroup of the permutation group $G_\sigma \subseteq \mathrm{Perm}(\mathrm{ar}(\sigma))$ on the tuples of $X^{\mathrm{ar}(\sigma)}$. The most well-known example is the 'bag' or 'multiset' functor which is given by the choice $\Sigma = \mathbb{N}$, $\mathrm{ar} = \mathrm{id}_{\mathbb{N}}$ and $G_n = \mathrm{Perm}(n), n \in \mathbb{N}$. Analytical functors preserve weak-pullbacks (in fact wide pullbacks, see [1]), and thus $R_{\mathsf{G}} = \leq_{\mathsf{G}}$.

▶ **Proposition 2.** *The posetification of an analytic functor* $\mathsf{G}_\Sigma : \mathbf{Set} \to \mathbf{Set}$ *is given by* $\mathsf{G}'_\Sigma X = (\mathsf{G}_\Sigma\mathsf{V}X, \overline{\mathsf{G}_\Sigma}\leq)$.

Proof. To simplify the notation we assume that each $A_\sigma = 1$, fix an element σ with arity $\mathrm{ar}(\sigma) = n$, and denote by $[(x_1, \ldots, x_n)]$ the equivalence class of the tuple $(x_1, \ldots, x_n)$ under the action of G_σ. Note that by definition of G_Σ, two elements of $\mathsf{G}_\Sigma\mathsf{V}X$ can only be related by $\leq_{\mathsf{G}}$ if they belong to the same σ-component of the coproduct. Moreover, if $[(x_1, \ldots, x_n)] \leq_{\mathsf{G}} [(y_1, \ldots, y_n)]$, then by definition there exists a permutation $\pi \in G_\sigma$ such that $(x_1, \ldots, x_n) \leq (y_{\pi(1)}, \ldots, y_{\pi(n)})$ (where $\leq$ here is component-pointwise). Similarly, if $[(y_1, \ldots, y_n)] \leq_{\mathsf{G}} [(x_1, \ldots, x_n)]$, there exists a permutation $\rho \in G_\sigma$ such that $(y_1, \ldots, y_n) \leq (x_{\rho(1)}, \ldots, x_{\rho(n)})$. It follows that $(x_1, \ldots, x_n) \leq (x_{\pi(\rho(1))}, \ldots, x_{\pi(\rho(n))})$, and since $\pi\rho$ is of finite order we easily get by iterating at most n times that $(x_1, \ldots, x_n) = (x_{\pi(\rho(1))}, \ldots, x_{\pi(\rho(n))})$. This in turn implies that $(y_{\pi(1)}, \ldots, y_{\pi(n)}) \leq (x_1, \ldots, x_n)$, from which we can conclude that $[(x_1, \ldots, x_n)] = [(y_1, \ldots, y_n)]$. Thus $\leq_{\mathsf{G}} = \overline{\mathsf{G}_\Sigma}\leq$ is anti-symmetric. ◀

In particular the posetification of the bag functor B is given by $\mathsf{B}'(X, \leq) = (\mathsf{B}\mathsf{V}X, \overline{\mathsf{B}}\leq)$.

3.3 Posetification of the monotone neighbourhood functor M

Recall that the *monotone neighbourhood functor* $\mathsf{M} : \mathbf{Set} \to \mathbf{Set}$ if defined on sets by

$$\mathsf{M}X = \{A \subseteq \mathcal{P}X \mid U \in A, U \subseteq V \Rightarrow V \in A\}$$

and on functions $f : X \to Y$ by taking the double inverse image $(f^{-1})^{-1} : \mathsf{M}X \to \mathsf{M}Y$. It is not hard to check that $\mathsf{M}f$ can be described more simply as $\mathsf{M}f(A) = \uparrow f[A]$, where $\uparrow f[A]$ is the upward closure (under inclusion) of the direct image of A by f. With this in place we can compute the coinserter (9)

$$\mathsf{DMV}X^2 \underset{\uparrow\pi_0[-]}{\overset{\uparrow\pi_1[-]}{\rightrightarrows}} \mathsf{DMV}X \xrightarrow{e_X} \mathsf{M}'X.$$

This time we need to consider the relation $R_{\mathsf{M}} \subseteq \mathsf{MV}X \times \mathsf{MV}X$ defined by

$$(A, B) \in R_{\mathsf{M}} \Leftrightarrow \exists C \in \mathsf{MV}X^2.\ \uparrow\pi_0[C] = A \ \&\ \uparrow\pi_1[C] = B$$

It is known (see [11]) that M does *not* preserve weak pullbacks, and in particular we cannot assume that the relation R_M is transitive. The proof of the following result can essentially be found in Theorem 8.25 of [10][3]

▶ **Proposition 3.** $(A, B) \in \leq_{\mathsf{M}}$ *iff* $\forall a \in A.\exists b \in B.\ \uparrow b \subseteq \uparrow a$ *and* $\forall b \in B.\exists a \in A.\ \downarrow a \subseteq \downarrow b$.

For any $A \in \mathsf{MV}X$ and $a \in A$, we write $\uparrow a$ for the upward closure of a under the order of X, and $\downarrow(\uparrow(A))$ for the set $\downarrow\{\uparrow a \mid a \in A\}$, where the downward closure is taken with respect to the inclusion. The following corollaries are then easy to check.

▶ **Corollary 4.** $A \equiv_{\mathsf{M}} B$ *iff* $\downarrow(\uparrow(A)) = \downarrow(\uparrow(B))$, *and moreover* $A \equiv_{\mathsf{M}} \downarrow(\uparrow(A))$.

▶ **Corollary 5.** *The posetification* M *is given by* $\mathsf{M}'X = (\mathsf{Down}(\mathsf{Up}(X)), \leq_{\mathsf{M}})$.

3.4 Posetification of the neighbourhood functor N

Consider the adjunctions $\mathsf{F}_{\mathsf{BA}} \dashv \mathsf{U}_{\mathsf{BA}} : \mathbf{BA} \to \mathbf{Set}$ and $\mathsf{F}_{\mathsf{CABA}} \dashv \mathsf{U}_{\mathsf{CABA}} : \mathbf{CABA} \to \mathbf{Set}$. The monad $\mathsf{U}_{\mathsf{BA}}\mathsf{F}_{\mathsf{BA}}$ is naturally isomorphic to the finitary version N_f of the neighbourhood functor N. A natural isomorphism is given by the natural transformation $\alpha : \mathsf{N}_f \to \mathsf{U}_{\mathbf{BA}}\mathsf{F}_{\mathbf{BA}}$ given at each X by

$$\alpha_X(A) = \bigvee\{\bigwedge a \wedge \bigwedge(a)^c \mid a \in A\}$$

which is indeed a boolean term since A and each $a \in A$ are finite. The inverse of α is built as follows: given a boolean term over X in conjunctive normal form, check for each clause $\bigwedge_{p\in a_1} p \wedge \bigwedge_{q\in a_2} \neg q$ if $a_1 \cup a_2 = X$, if not rewrite the clause as the equivalent CNF expression

$$\bigvee\{\bigwedge_{p\in a_1\cup a_3} p \ \wedge \bigwedge_{q\in a_2\cup(a_3)^c} \neg q \mid a_3 \subseteq X \setminus (a_1 \cup a_2)\}$$

This yields a finite disjunction $\bigvee\{\bigwedge_{p\in a_i} p \wedge \bigwedge_{q\notin a_i} \neg q \mid i \in I\}$, which we associate with $\{a_i\}_{i\in I} \in \mathsf{N}_f X$. Similarly, the monad $\mathsf{U}_{\mathsf{CABA}}\mathsf{F}_{\mathsf{CABA}}$ is naturally isomorphic to the full neighbourhood functor N. We can use the special properties of the adjunctions above to compute the posetification of N_f and N indirectly, but relatively straightforwardly.

[3] We thank Clemens Kupke for pointing out this reference.

▶ **Proposition 6.** *Let X be a poset presented by the coinserter $\mathsf{DV}X^{\mathbb{2}} \rightrightarrows \mathsf{DV}X \xrightarrow{c'} X$ and let $\mathsf{F} \dashv \mathsf{U}$ denote either of the adjunctions $\mathsf{F}_{\mathsf{BA}} \dashv \mathsf{U}_{\mathsf{BA}}$ or $\mathsf{F}_{\mathsf{CABA}} \dashv \mathsf{U}_{\mathsf{CABA}}$, then the coinserter of $\mathsf{FV}X^{\mathbb{2}} \rightrightarrows \mathsf{FV}X$ is $\mathsf{F}c : \mathsf{FV}X \to \mathsf{FC}X$, where $c : \mathsf{V}X \to \mathsf{C}X$ is the adjoint of c'.*

Proof. Note first that c coequalizes $\pi_0, \pi_1 : \mathsf{V}X^{\mathbb{2}} \rightrightarrows \mathsf{V}X$; indeed two elements x, y lie in the same connected component precisely when $x \leq y$ or $y \leq x$. It follows that Fc coequalizes $\mathsf{FV}X^{\mathbb{2}} \rightrightarrows \mathsf{FV}X$, and in particular $Fc \circ \pi_0 \leq Fc \circ \pi_1$. We need to show that it is in fact a coinserter for which, due to Prop.1, it is enough to show that it is a coequaliser. Let $d : \mathsf{FV}X \to Y$ with $d \circ \mathsf{F}\pi_0 = d \circ \mathsf{F}\pi_1$. Let $d' : \mathsf{V}X \to \mathsf{U}Y$ be the adjoint transpose of d. We have that d' factors through c. Writing $\eta : \mathsf{Id} \to \mathsf{UF}$ for the unit of the adjunction, it follows that there is a unique $f : \mathsf{FC}X \to Y$ such that $\mathsf{U}f \circ \eta_{CX} \circ c = d'$, or, equivalently, that $f \circ \mathsf{F}c = d$. We have shown that $\mathsf{F}c$ is the coequaliser (and coinserter) of $\mathsf{FV}X^{\mathbb{2}} \rightrightarrows \mathsf{FV}X$. ◀

▶ **Lemma 7.** *U_{BA} and $\mathsf{U}_{\mathsf{CABA}}$ preserve reflexive coequalisers.*

Proof. Being a variety of finitary algebras, U_{BA} preserves sifted colimits and, in particular, reflexive coequalisers [2]. In the case of $\mathsf{U}_{\mathsf{CABA}}$ we use that **CABA** is equivalent to $\mathbf{Set}^{\mathrm{op}}$ and that $[-, 2] : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Set}$ preserves reflexive coequalisers, see [4, 5.1.5 Lemma]. ◀

▶ **Theorem 8.** *The posetification of N_f is $\mathsf{DN}_f\mathsf{C}$ and the posetification of N is DNC.*

Proof. We use the notation of Proposition 6. It follows from (3) that D preserves all ordinary colimits and, in particular, reflexive coequalizers. Due to the lemma U preserves reflexive coequalizers. Like all functors, D and U preserve the symmetry of coinserters. It follows from Proposition 6 that

$$\mathsf{DUFV}X^{\mathbb{2}} \underset{\mathsf{DUF}\pi_0}{\overset{\mathsf{DUF}\pi_1}{\rightrightarrows}} \mathsf{DUFV}X^{\mathbb{2}} \longrightarrow \mathsf{DUFC}X$$

is a coequalizer and coinserter. Thus DUFC is the posetification of UF. ◀

▶ **Remark 9.** This result is curious at first sight. Due to (3) and (7), $\mathsf{DN}_f\mathsf{C}$ and DNC are also the *right* Kan-extensions of N_f and N, respectively. To better understand the situation let us recall that we need the posetification to be locally monotone, which means that it must be an *enriched* left Kan extension. Now, working in the ordered setting (ie **Pos**-enriched), Prop.1 enforces that Boolean algebras cannot be quotiented by a partial order in **BA** without quotienting by its symmetric closure.

For example, let $X = \{p < q\}$, so that $\mathsf{V}X = \{p, q\}$ and $\mathsf{V}X^{\mathbb{2}} = \{(p,p), (p,q), (q,q)\}$. Then dividing $\mathsf{FV}X$ by $p \leq q$ 'equationally' gives the Boolean algebra 2^3 whereas the coinserter gives $\mathsf{F}1 = 2^2$. In more detail: Dividing $\mathsf{FV}X$ by $p \leq q$ (or $p \wedge q = p$ or $p \wedge \neg q = 0$) gives the Boolean algebra 2^3 because $\mathsf{FV}X = 2^4$ and $p \leq q$ kills one of the 4 atoms, namely $p \wedge \neg q$. On the other hand, the coinserter divides $\mathsf{FV}X$ by a larger theory, namely by one in which negation is monotone. (Recall that in the **Pos**-enriched setting all operations are monotone. Of course, one can still have algebras with "non-monotone" operations like negation in **BA**, but then the **Pos**-enriched order must be discrete. Which does not prevent us from recovering the natural order of **BA**s by considering **BA** as a subcategory of **DL**.)

4 Positivication

As mentioned in the introduction, a boolean coalgebraic logic is given by an endofunctor $T : \mathbf{Set} \to \mathbf{Set}$ determining the type of coalgebraic semantics, and an endofunctor $L :$

BA $\to$ **BA** constructing general 'modal algebras'. We have just seen how to extend a functor $T : \mathbf{Set} \to \mathbf{Set}$ to a functor $T' : \mathbf{Pos} \to \mathbf{Pos}$ to extend a type of coalgebraic semantics to posets. Now we want to extend a boolean syntax-building functor to a positive syntax-building functor $L' : \mathbf{DL} \to \mathbf{DL}$.

An obvious idea is to work dually to the posetification procedure and define the *positivication* of $L : \mathbf{BA} \to \mathbf{BA}$ as the right Kan extension $L' = \mathrm{Ran}_{\mathsf{W}}(\mathsf{W}L)$, where W is the inclusion $\mathbf{BA} \to \mathbf{DL}$.[4] Note that this gives us what we would expect: (i) an isomorphism $\beta : L'\mathsf{W} \cong \mathsf{W}L$, saying that L' is the same as L on boolean algebras and (ii) for all $H : \mathbf{DL} \to \mathbf{DL}$ and all $\alpha : H\mathsf{W} \to \mathsf{W}L$ there is a unique $\gamma : H \to L'$ such that $\beta \circ \gamma = \alpha$, saying that L' is the optimal (or co-universal) extension with (i).

It is also worth emphasising that $\beta : L'\mathsf{W} \cong \mathsf{W}L$ will be doing some real work once the abstract framework is instantiated with concrete examples. In particular, β^{-1} will translate a boolean formula ϕ in LB into a positive formula $\beta_B^{-1}(\phi)$ where negation is eliminated from the modal part and pushed "onto the atoms in B".

In order to capture this process of eliminating negation in the abstract categorical framework, we need to understand, once again, the Kan extension in the **Pos**-enriched way. To compute these right Kan extensions we use a presentation of distributive lattices which will play the same role in the computation of positivications as (6) played in the case of posetifications.

▶ **Proposition 10.** *Every $A \in \mathbf{DL}$ is the inserter of a diagram of boolean algebras (where in_1, in_2 are the canonical embeddings and e is the unit at A of the adjunction $\mathsf{G} \dashv \mathsf{W}_o$):*

$$A \xrightarrow{\ e\ } \mathsf{WG}A \underset{\mathsf{WG}in_1}{\overset{\mathsf{WG}in_2}{\rightrightarrows}} \mathsf{WG}(\mathbb{2} \bullet A) \qquad (10)$$

▶ **Remark 11.** **1.** The tensor $\mathbb{2} \bullet A$ is isomorphic to $A + A$ modulo inequations $in_1 a \leq in_2 a$. If $a \in A$ has a complement then $in_1 a = in_2 a$. If all elements of A are complemented, that is, if the distributive lattice A happens to be a boolean algebra, then $\mathbb{2} \bullet A \cong A$.

2. Equivalently, $\mathbb{2} \bullet A$ can be represented as the distributive lattice generated by $\{\Box_1 a \mid a \in A\}$ and $\{\Box_2 a \mid a \in A\}$ modulo equations specifying that $\Box_1, \Box_2$ preserve all **DL**-operations and modulo inequations $\Box_1 a \leq \Box_2 a$.

3. Let A^∂ be the Priestley space dual to $A \in \mathbf{DL}$, that is, the space of prime filters on A. Then $2 \bullet A$ is dual to $(A^\partial)^{\mathbb{2}}$. [5]

4. The inserters (10) are reflexive. This follows easily from the definition of tensor with $\mathbb{2}$ as $\mathcal{A}(\mathbb{2} \bullet A, A') \cong [\mathbb{2}, \mathcal{A}(A, A')]$ giving us a half-inverse $\mathbb{2} \bullet A \to A$ of both $in_1, in_2 : A \to \mathbb{2} \bullet A$ as the transpose of the map $\mathbb{2} \to \mathcal{A}(A, A)$ which maps both truth values to id_A.

Proof of Prop.10. Let $(-)^\partial$ be the functor that dualises **DL**s to Priestley spaces. Its composition with the forgetful functor to posets we denote by $\mathsf{S}' : \mathbf{DL} \to \mathbf{Pos}$. Applying it to (10) yields a reflexive coinserter in **Pos**

$$\mathsf{S}'A \longleftarrow \mathsf{DV}(\mathsf{S}'A) \underset{\longleftarrow}{\overset{\longleftarrow}{\uparrow}} \mathsf{DV}(\mathsf{S}'A)^{\mathbb{2}} \qquad (11)$$

as in (5). Now it only remains to check that it is also a coinserter in Priestley spaces, from which the result follows by duality. ◀

[4] W is fully faithful. Moreover, whereas $B \in \mathbf{BA}$ is discrete (see Section 2.2), $\mathsf{W}B \in \mathbf{DL}$ is equipped with its natural order. So while W 'forgets negation' it also 'adds the order'.

[5] Cotensors in Priestley spaces are computed as cotensors in **Pos**, since the forgetful functor from Priestely spaces to posets preserves and creates all **Pos**-enriched limits.

The following theorem requires some knowledge of enriched category theory. Even though the theorem is one of the main contributions, we encourage the reader so inclined to skip directly ahead to its corollary, which is all that is needed to follow the rest of the paper.

▶ **Theorem 12.** $\mathsf{W} : \mathbf{BA} \to \mathbf{DL}$ *is co-dense and the inserters* (10) *form a co-density presentation in the sense of [16, Thm 5.19].*

Together with [16, Thm 5.30] we obtain

▶ **Corollary 13.** *Every* $\mathbf{BA}$*-endofunctor* L *has a positivication* L' *which can be computed explicitly at any distributive lattice* A *as the inserter*

$$L'A \longrightarrow \mathsf{W}LG A \underset{\mathsf{W}LG in_1}{\overset{\mathsf{W}LG in_2}{\uparrow}} \mathsf{W}LG(\mathbb{2} \bullet A) \tag{12}$$

Moreover, a functor is a positivication iff it preserves Boolean algebras and the inserters (10)*.*

▶ **Proposition 14.** *If* $L : \mathbf{BA} \to \mathbf{BA}$ *is finitary, then so is its positivication* $L' : \mathbf{DL} \to \mathbf{DL}$*.*

Proof. All operations involved in (12), that is, W, L, G, $\mathbb{2} \bullet -$, preserve filtered colimits. And filtered colimits commute with finite weighted limits, see [15, Prop.4.9].[6] ◀

4.1 Positivication of normal modal logic

Of course, the positivication of Kripke's normal modal logic with one meet-preserving $\Box$ should turn out to be Dunn's positive modal logic [8]. We show this in a roundabout way which has the advantage of making precise the relationship of our notion of positivication with the procedure employed in [3].

To summarise, going back to Diagrams (1) and (2), [3] starts with T and then, on the one hand define L via $LB = \mathsf{P}T\mathsf{S}B$ on finite $\mathbf{BA}$s and, on the other hand, define L' via $L'A = \mathsf{P}'T'\mathsf{S}'A$ on finite $\mathbf{DL}$s with T' the posetification of T.

▶ **Theorem 15.** *Let* $T : \mathbf{Set} \to \mathbf{Set}$ *preserve finite sets and let* L *be given on finite* $B \in \mathbf{BA}$ *by* $LB = \mathsf{P}T\mathsf{S}B$*. Let* T' *be the posetification of* T *and let* L' *be given on finite* $A \in \mathbf{DL}$ *by* $L'A = \mathsf{P}'T'\mathsf{S}'A$*. Then* L' *is the positivication of* L*.*

Proof. We have to show that $L' = \mathrm{Ran}_W WL$. By duality and definition of T' as $\mathrm{Lan}_D DT$, we know that L' and $\mathrm{Ran}_W WL$ agree on finite $\mathbf{DL}$s. Now the claim follows from Prop.14. ◀

▶ **Remark 16.**

1. The conditions of the theorem are not strong enough to guarantee that L' is strongly finitary and thus has a presentation by operations and equations. As shown in [3, Thm 6.20], this is the case if T preserves weak pullbacks.
2. In the case of graded modal logic, L is different from $\mathsf{P}T\mathsf{S}$ even on finite $\mathbf{BA}$s. Therefore, the approach of [3] cannot be applied. We leave a description of the positivication of graded modal logic for a sequel.

Now, if, in the notation of the theorem, we start with T as the powerset functor, it is well known that L is Kripke's normal modal logic and [3] shows that L' is Dunn's positive modal logic. It follows from Theorem 15, that the latter is indeed the positivication of the former.

[6] We are grateful to John Power for pointing out this reference.

4.2 Positivication of non-monotone modal logics

The basic idea of positivication is the following. Given a modal logic with monotone modalities, add the duals and find the axioms so that boolean negation can be pushed to the atoms. But our abstract definition of positivication is powerful enough to also apply to logics which have modalities that are not monotone.

We will study what happens in such a situation through the example of the modal logic with one $\Box$ that does not obey any equations, not even monotonicity. In our functorial setting, this logic is given by $L = \mathsf{F}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}} : \mathbf{BA} \to \mathbf{BA}$. Recall the functors $\mathsf{W}, \mathsf{K}, \mathsf{G}$ from (4).

▶ **Theorem 17.** *The positivication of* $\mathsf{F}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}} : \mathbf{BA} \to \mathbf{BA}$ *is* $\mathsf{WF}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}}\mathsf{K} : \mathbf{DL} \to \mathbf{DL}$.

Proof. We know from Theorem 8 that $\mathrm{Lan}_{\mathsf{D}}\mathsf{DU}_{\mathsf{BA}}\mathsf{F}_{\mathsf{BA}} = \mathsf{DU}_{\mathsf{BA}}\mathsf{F}_{\mathsf{BA}}\mathsf{C}$. By duality, on finite **DL**s, $\mathrm{Ran}_{\mathsf{W}}\mathsf{WF}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}} = \mathsf{WF}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}}\mathsf{K}$. Now the result follows from Prop.14 since all of $\mathsf{W}, \mathsf{F}_{\mathsf{BA}}, \mathsf{U}_{\mathsf{BA}}, \mathsf{K}$ are finitary. ◀

▶ Remark. From a logical point of view, the appearance of $\mathsf{K} : \mathbf{DL} \to \mathbf{BA}$ in Thm 17 tells us that, given $A \in \mathbf{DL}$, we are only allowed to build a formula $\Box a$, $a \in A$, *if* a *lies in a boolean subalgebra* (ie a has a complement). This side condition takes us out of the realm of equational logic and, hence, of modal logics given by axioms. This is related to the fact that K is *not* strongly finitary [3, Example 6.6] and, therefore, functors involving K cannot be expected to have a presentation by operations and equations.

To give another example of an extension by non-monotone modalities, the logic $\mathsf{W}_o\mathsf{G} : \mathbf{DL}_o \to \mathbf{DL}_o$ is a modal logic over distributive lattices with one unary modality obeying the axioms of negation. In other words, $\mathsf{W}_o\mathsf{G}$-algebras over $\mathbf{DL}_o$ are just boolean algebras. Clearly, $\mathsf{W}_o\mathsf{G}$ is not locally monotone and negation, considered as a unary modality, cannot be 'positivised'. Nevertheless, writing I for the inclusion $\mathbf{DL}_o \to \mathbf{DL}$, the right Kan extension $\mathrm{Ran}_I I\mathsf{W}_o\mathsf{G}$ does exist and is the identity.

▶ **Proposition 18.** $\mathrm{Ran}_I I\mathsf{W}_o\mathsf{G} = \mathsf{Id}$.

Proof. Going back to (8), we have $\tilde{H} = \mathsf{W}_o\mathsf{G}$, which means that we can take $H = \mathsf{Id}$. But then, by Thm 12 and [16, Thm 5.1], we have $H' = \mathsf{Id}$. ◀

To summarise, we have seen two examples of positivication of modal extensions by non-monotone modalities. In the first case, the non-monotone modality was made monotone by adding a side-condition restricting its use. In the second case, the non-monotone modality was eliminated.

5 Positive coalgebraic logic

5.1 Semantics

Recall from the introduction that we wish to move from an ordinary BCL given by the diagram (1) to a **Pos**-enriched PCL given by diagram (2). In Sections 3 and 4 we have shown how to build T' from T and L' from L respectively. The missing element is the construction of δ' from δ. Let us first remind the reader of how δ defines the interpretation, this will also be the occasion to fix some notation.

▶ **Theorem 19** ([21])**.** *An endofunctor* L *on a variety* $\mathcal{A}$ *has a finitary presentation by operations and equations iff it preserves sifted colimits, in which case* $\mathrm{Alg}(L)$ *is a variety.*

For L strongly finitary on either **BA** or **DL** and A an object of the corresponding category, let $\mathsf{F}_L(A)$ denote the free L-algebra over A. In particular, if V denotes a countable set of propositional variables and $\mathsf{F}V$ is the freely generated object over V in **BA** or **DL**, then the free L-algebra $\mathsf{F}_L(\mathsf{F}V)$ is the algebra of L-modal formulas for the syntax defined by L, which we denote more succinctly by $\mathcal{L}$. Now, let $\gamma : X \to TX$ be a T-coalgebra and assume that it comes equipped with a *valuation* $v : \mathsf{F}V \to \mathsf{P}X$, the interpretation map $[\![-]\!]_{(\gamma,v)}$ is the unique map given by initiality of $\mathcal{L}$ amongst $L(-) + \mathsf{F}V$-algebras.

$$\begin{array}{ccccc} L\mathcal{L} + \mathsf{F}V & & \xrightarrow{\qquad\qquad\simeq\qquad\qquad} & & \mathcal{L} \\ {\scriptstyle L[\![-]\!]_{(\gamma,v)}+\mathrm{id}_{\mathsf{F}V}}\downarrow & & & & \downarrow{\scriptstyle [\![-]\!]_{(\gamma,v)}} \\ LPX + \mathsf{F}V & \xrightarrow{\delta_X+\mathrm{id}_{\mathsf{F}V}} & \mathsf{P}TX + \mathsf{F}V & \xrightarrow{\mathsf{P}\gamma+v} & \mathsf{P}X \end{array}$$

We can now turn to defining δ' from δ. To avoid unsightly $(-)^{\mathrm{op}}$ symbols appearing everywhere we simply consider $\mathsf{P}, \mathsf{P}', \mathsf{S}, \mathsf{S}'$ to be *contravariant functors* throughout (as opposed to covariant functors from/to an $(-)^{\mathrm{op}}$ category). The following definition was given in [3].

▶ **Definition 20.** A logic (L', δ') for T' is *a positive fragment* of the logic (L, δ) for T, if there exist natural transformations $\alpha : T'\mathsf{D} \to \mathsf{D}T$ and $\beta : L'\mathsf{W} \to \mathsf{W}L$ such that $\mathsf{W}\delta \circ \beta\mathsf{S} = \mathsf{S}'\alpha \circ \delta'\mathsf{D}$.

Clearly, we have natural transformations $\alpha : T'\mathsf{D} \to \mathsf{D}T$ and $\beta : L'\mathsf{W} \to \mathsf{W}L$ by construction of the posetification T' and of the positivication L'. We can construct a natural transformation δ' as follows. First, it is not hard to check that $\mathsf{W} \circ \mathsf{P} = \mathsf{P}' \circ \mathsf{D}$. Thus, given a natural transformation $\delta : L\mathsf{P} \to \mathsf{P}T$ we get a natural transformation

$$L'\mathsf{P}'\mathsf{D} = L'\mathsf{W}\mathsf{P} \overset{\beta_{\mathsf{P}}}{\Longrightarrow} \mathsf{W}L\mathsf{P} \overset{W\delta}{\Longrightarrow} W\mathsf{P}T = \mathsf{P}'\mathsf{D}T$$

▶ **Lemma 21.** *For any poset X, the following diagram is an inserter:*

$$\mathsf{P}'T'X \longrightarrow \mathsf{P}'\mathsf{D}T\mathsf{V}X \underset{\mathsf{P}'\mathsf{D}T\pi_0}{\overset{\mathsf{P}'\mathsf{D}T\pi_1}{\rightrightarrows}} \mathsf{P}'\mathsf{D}T\mathsf{V}X^{\mathbb{2}}$$

Proof. $\mathsf{D}TX_0 \rightrightarrows \mathsf{D}TX_1 \to T'X$ is a coinserter and since P' is the enriched hom functor $\hom(-, \mathbb{2})$ it turns coinserters into inserters. ◀

By naturality of β and δ, the two right-hand side squares of the following diagram commute, and this defines a **Pos**-enriched natural transformation $\delta' : L'\mathsf{P}' \to \mathsf{P}'(T')$.

$$\begin{array}{ccccc} L'\mathsf{P}'X & \longrightarrow & L'\mathsf{P}'\mathsf{D}\mathsf{V}X & \underset{L'\mathsf{P}'\mathsf{D}\pi_0}{\overset{L'\mathsf{P}'\mathsf{D}\pi_1}{\rightrightarrows}} & L'\mathsf{P}'\mathsf{D}\mathsf{V}X^{\mathbb{2}} \\ {\scriptstyle \delta'_X}\downarrow & & \downarrow{\scriptstyle \mathsf{W}\delta_{\mathsf{V}X} \circ \beta_{\mathsf{P}\mathsf{V}X}} & & \downarrow{\scriptstyle \mathsf{W}\delta_{\mathsf{V}X^{\mathbb{2}}} \circ \beta_{\mathsf{P}\mathsf{V}X^{\mathbb{2}}}} \\ \mathsf{P}'T'X & \longrightarrow & \mathsf{P}'\mathsf{D}T\mathsf{V}X & \underset{\mathsf{P}'\mathsf{D}T\pi_0}{\overset{\mathsf{P}'\mathsf{D}T\pi_1}{\rightrightarrows}} & \mathsf{P}'\mathsf{D}T\mathsf{V}X^{\mathbb{2}} \end{array} \tag{13}$$

▶ **Theorem 22.** *With δ' defined as above, the logic (L', δ') for T' is a positive fragment of the logic (L, δ) for L'.*

Proof. We need to check that $\mathsf{W}\delta \circ \beta\mathsf{S} = \mathsf{S}'\alpha \circ \delta'\mathsf{D}$. Given a set X, the poset $\mathsf{D}X$ has a completely trivial coinserter presentation given by $\mathsf{D}X \rightrightarrows \mathsf{D}X \to \mathsf{D}X$, and in particular $T'\mathsf{D}X = \mathsf{D}TX$, i.e. $\alpha_X = \mathrm{id}_X$, and the result follows from the diagram (13). ◀

5.2 Completeness

We say that a BCL or a PCL (L, δ) is *weakly complete* for T-coalgebras if for any formulas $\phi, \psi \in \mathcal{L}$ such that $\phi \not\leq \psi$, there exists a T-coalgebra $\gamma : X \to TX$ and a valuation $v : \mathsf{F}_{\mathsf{BA}} V \to \mathsf{P}X$ ($v : \mathsf{F}_{\mathsf{DL}} V \to \mathsf{P}'X$ for posets) and an element $x \in X$ such that $x \in [\![\phi]\!]_{(\gamma,v)}$ but $x \notin [\![\psi]\!]_{(\gamma,v)}$. The following theorem gives a sufficient condition for weak completeness.

▶ **Theorem 23** ([17]). *A BCL or a PCL (L, δ) for a functor T is weakly complete if δ is component-wise injective.*

Weak completeness transfers from a boolean logic to its positive fragment.

▶ **Theorem 24.** *For a BCL (L, δ) defined by a strongly finitary functor $L : \mathbf{BA} \to \mathbf{BA}$, if δ is component-wise injective, then so is δ'. In particular (L', δ') is then weakly complete.*

Proof. Recall first that finitary (and thus strongly finitary) functors $L : \mathbf{BA} \to \mathbf{BA}$ preserve injective maps ([20, Lemma 6.14]). Since the natural transformation $\beta : L'\mathsf{W} \to \mathsf{W}L$ is an isomorphism, it follows that the vertical legs of (13) are injective. Since $\mathsf{P}' = (-, \mathbb{2})$ turns coinserters into inserters and L' preserves inserters by construction, the top row of (13) is an inserter as well, and hence injective. It follows that δ'_X must be injective. ◀

The case of normal modal logic. Let $L : \mathbf{BA} \to \mathbf{BA}$ be the syntax-building functor for normal modal logic:

$$LA = (\mathsf{F}_{\mathsf{BA}}\mathsf{U}_{\mathsf{BA}}A)/\{\Diamond(a \vee b) = \Diamond a \vee \Diamond b, \Diamond\bot = \bot\}$$

and let $\delta : L\mathsf{P} \to \mathsf{P}\mathcal{P}$ be the semantic transformation for normal modal logic, i.e.

$$\delta_X(\Diamond U) = \{V \subseteq X \mid V \cap U \neq \emptyset\}$$

We have computed the posetification P' or P in Section 3 and shown that the positivication L' of L is given by Dunn's syntax ([8]) in Section 4. The following result is well-known, and can be shown directly.

▶ **Theorem 25.** *The natural transformation $\delta : L\mathsf{P} \to \mathsf{P}\mathcal{P}$ is component-wise injective.*

▶ **Corollary 26.** *If L' is the positivication of the syntax functor for normal modal logic, $\mathcal{P}'$ the posetification of the powerset functor, and $\delta' : L'\mathsf{P}' \to \mathsf{P}'\mathcal{P}'$ the semantics generated from $\delta : L\mathsf{P} \to \mathsf{P}\mathcal{P}$ by diagram (13), then the PCL (L', δ') is weakly complete for $\mathcal{P}'$-coalgebras.*

6 Conclusion and future work

We have presented positive coalgebraic logic at the same level of generality as boolean coalgebraic logic, and developed a method by which boolean coalgebraic logics can systematically be turned into positive coalgebraic logics. We have also shown that completeness follows automatically from the boolean case in this setup. More broadly, we have also presented a practical application of enriched category theory in logic by showing that positive modal logic amounts to a type of **Pos**-enriched coalgebraic logic. We believe that this perspective offers a deep insight into the fundamental difference between boolean and positive logics.

Much remains to be investigated. First, we do not yet have much practical experience and tools to compute positivications. As Section 4 illustrates, our calculations are all indirect. In particular we would like to compute the positivication of graded modal logic. On the logic side, we have good reasons to believe that strong completeness transfers from the boolean to

the positive case for a large class of functors. On the other hand, we believe that expressivity does not transfer in general. All this will be investigated in a future companion publication to this work.

References

1 J. Adámek, H. P. Gumm, and V. Trnková. Presentation of set functors: a coalgebraic perspective. *Journal of Logic and Computation*, 20(5):991–1015, 2010.

2 J. Adámek, J. Rosickỳ, and E. M. Vitale. *Algebraic theories: a categorical introduction to general algebra*, volume 184. Cambridge University Press, 2010.

3 A. Balan, A. Kurz, and J. Velebil. Positive fragments of coalgebraic logics. In *International Conference on Algebra and Coalgebra in Computer Science*, pages 51–65. Springer, 2013.

4 M. Barr and C. Wells. *Toposes, triples and theories*, volume 278. Springer-Verlag, 1985.

5 S. L. Bloom and J. B. Wright. P-varieties-a signature independent characterization of varieties of ordered algebras. *Journal of Pure and Applied Algebra*, 29(1):13–58, 1983.

6 S. Celani and R. Jansana. A new semantics for positive modal logic. *Notre Dame Journal of Formal Logic*, 38(1), 1997.

7 C. Cirstea, A. Kurz, D. Pattinson, L. Schröder, and Y. Venema. Modal logics are coalgebraic. *The Computer Journal*, 54(1):31–41, 2009.

8 J. M. Dunn. Positive modal logic. *Studia Logica*, 55(2):301–317, 1995.

9 M. Gehrke, H. Nagahashi, and Y. Venema. A Sahlqvist theorem for distributive modal logic. *Annals of pure and applied logic*, 131(1):65–102, 2005.

10 H. H. Hansen. Monotonic modal logics. Technical Report PP-2003-24, ILLC, University of Amsterdam, 2003.

11 H. H. Hansen and C. Kupke. A coalgebraic perspective on monotone modal logic. *Electronic Notes in Theoretical Computer Science*, 106:121–143, 2004.

12 B. Jacobs and A. Sokolova. Exemplaric expressivity of modal logics. *J. Log. and Comput.*, 20:1041–1068, October 2010.

13 André Joyal. Foncteurs analytiques et especes de structures. *Combinatoire énumérative*, pages 126–159, 1986.

14 K. Kapulkin, A. Kurz, and J. Velebil. Expressiveness of positive coalgebraic logic. *Advances in Modal Logic*, 9:368–385, 2012.

15 G. M. Kelly. Structures defined by finite limits in the enriched context, i. *Cahiers de topologie et géométrie différentielle catégoriques*, 23(1):3–42, 1982.

16 M. Kelly. *Basic concepts of enriched category theory*, volume 64. CUP Archive, 1982.

17 C. Kupke, A. Kurz, and D. Pattinson. Algebraic semantics for coalgebraic logics. In *CMCS 2004*, volume 106 of *ENTCS*, pages 219–241, 2004.

18 C. Kupke, A. Kurz, and D. Pattinson. Ultrafilter Extensions for Coalgebras. In *CALCO 2005*, volume 3629, pages 263–277. Springer, 2005.

19 C. Kupke and D. Pattinson. Coalgebraic semantics of modal logics: an overview. *TCS*, 412(38):5070–5094, 2011. Special issue CMCS 2010.

20 A. Kurz and D. Petrişan. Presenting functors on many-sorted varieties and applications. *Information and Computation*, 208(12):1421–1446, 2010.

21 A. Kurz and J. Rosický. Strongly complete logics for coalgebras. *Logical Methods in Computer Science*, 8, 2012.

22 A. Kurz and J. Velebil. Quasivarieties and varieties of ordered algebras: regularity and exactness. *Mathematical Structures in Computer Science*, 2016.

23 L. Santocanale, Y. Venema, et al. Uniform interpolation for monotone modal logic. *Advances in Modal Logic*, 8:350–370, 2010.

Justified Sequences in String Diagrams: a Comparison Between Two Approaches to Concurrent Game Semantics

Clovis Eberhart[1] and Tom Hirschowitz[2]

1 **LAMA, CNRS, Université Savoie Mont Blanc, Bâtiment Le Chablais, Campus Scientifique, 73376 Le Bourget-du-Lac Cedex, France**
2 **LAMA, CNRS, Université Savoie Mont Blanc, Bâtiment Le Chablais, Campus Scientifique, 73376 Le Bourget-du-Lac Cedex, France**

Abstract

Recent developments of game semantics have given rise to new models of concurrent languages. On the one hand, an approach based on *string diagrams* has given models of CCS and the π-calculus, and on the other hand, Tsukada and Ong have designed a games model for a non-deterministic λ-calculus. There is an obvious, shallow relationship between the two approaches, as they both define innocent strategies as sheaves for a Grothendieck topology embedding "views" into "plays". However, the notions of views and plays differ greatly between the approaches: Tsukada and Ong use notions from standard game semantics, while the authors of this paper use string diagrams. We here aim to bridge this gap by showing that even though the notions of plays, views, and innocent strategies differ, it is mostly a matter of presentation.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases Concurrency, Sheaves, Presheaf models, Game Semantics

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.10

1 Introduction

1.1 Two approaches to concurrent game semantics

Game semantics [11], a branch of denotational semantics in which types are interpreted as some sort of games and programs as strategies in these games, has led to fully abstract models for a variety of functional languages. Recent advances in concurrent game semantics have produced new games models for a non-deterministic, simply-typed λ-calculus on the one hand [19], and for CCS and the π-calculus on the other hand [9, 10, 6]. These models are based on categories of innocent and concurrent strategies that are defined in both cases as categories of *sheaves* over a site of plays.

The first model, by Tsukada and Ong, has proven to successfully extend the most fundamental results of game semantics (interpreting terms as innocent strategies and composing strategies to form a CCC of arenas and innocent strategies) to a non-deterministic λ-calculus. The other approach, while arguably more complex and not as well developed, aims to give a general framework to build games models for different calculi, in order to study translations between them.

There is a clear, yet informal relationship between the two approaches in that they both define innocent strategies as sheaves for a Grothendieck topology induced by embedding *views* into *plays*. However, despite this similarity, the notions of views and plays differ significantly. Indeed, Tsukada and Ong [19] define them as *justified sequences* of moves satisfying

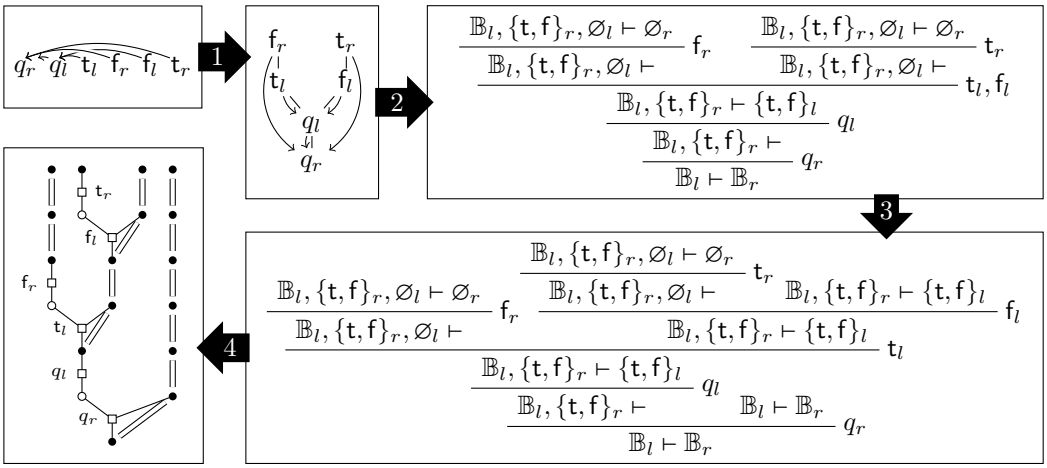


Figure 1 The big picture.

additional conditions, as in standard Hyland-Ong/Nickau (HON) game-semantics [11, 16], while in [9, 10, 6], plays are defined as *ad hoc* string diagrams describing the game, as originally suggested by Melliès in a different setting (*circa* 2008, published as [15]). Since the notions of plays differ significantly, it is legitimate to wonder to what degree the approaches are related.

1.2 The level of views and plays

In this paper, we go beyond this informal similarity and show a tight correspondence between the two approaches at the level of plays. Specifically, for each pair of *arenas* [11] A and B, we design categories $\mathbb{E}(A \vdash B)$ and $\mathbb{E}^{\mathbb{V}}(A \vdash B)$ respectively of plays and views for HON games, in the same spirit as those of the models for CCS and the π-calculus.

There are also standard categories $\mathbb{P}_{A,B}$ and $\mathbb{V}_{A,B}$ of plays and views as defined in standard game semantics (and which we call *HON-plays* and *HON-views* to disambiguate), with a notion of morphism inspired by Melliès [14]. We then embed these categories into $\mathbb{E}(A \vdash B)$ and $\mathbb{E}^{\mathbb{V}}(A \vdash B)$ respectively (whose objects we simply call *plays* and *views*).

We define this embedding using a third model, whose plays are proof trees in an *ad hoc* sequent calculus, and whose views are branches of those trees. The categories of trees and branches are equivalent to those of plays and views respectively, so they can be thought of as another possible representation of these objects. Trees may also be seen as a maximal parallelisation of HON-plays, while branches are simply equivalent to HON-views.

Let us show how this embedding works on an example, illustrated in Figure 1. HON-plays are based on the notion of *arena*:

▶ **Definition 1.** An *arena* is a simple forest, i.e., a directed, simple graph in which all vertices are uniquely reachable from a unique root (vertex without a parent).

Vertices are called *moves*, and roots deemed *initial*. A move m' is said to be *justified* by m when it is one of its children.

Let M_A be the set of moves of A, the *ownership* of any move $m \in M_A$ is O (for *Opponent*) if the length of the unique path from a root to m is even, and P (for *Proponent*) otherwise. So, e.g., all roots have ownership O. We denote this map $M_A \to \{P, O\}$ by λ_A.

▶ **Example 2.** The boolean arena $\mathbb{B}$ has a single root q, which is an Opponent move, and two Proponent moves t and f, both justified by q.

A HON-play on a pair of arenas (A, B) is a sequence of moves in $M_A + M_B$ that verifies some additional properties. The full definitions are recalled in Section 2. In our example, we study a play on $(\mathbb{B}_l, \mathbb{B}_r)$, where l and r are only present to show in which copy of $\mathbb{B}$ moves are played. We show how s may be mapped to a play P in our setting and P may conversely be mapped back to a HON-play isomorphic to s.

We here give a simplified description of our plays that should give the reader enough intuition to understand the mapping. The precise definitions can be found in Section 3.1. Our plays are based on the notion of *position*. Positions are sets of *players* that are labelled by sequents of arenas, either positive sequents ($\Gamma \vdash$) or negative ones ($\Gamma \vdash A$). In the first case, players are depicted $\circ$, and $\bullet$ in the second case. Positions may evolve over time according to the *moves* the different players play. Each type of player can only do the following: positive players $(\Gamma, A, \Delta \vdash)$ may turn into negative players $(\Gamma, A, \Delta \vdash A \cdot m)$ for any root m of A, and negative players $(\Gamma \vdash A)$ may spawn a positive player $(\Gamma, A \cdot m \vdash)$ for any root m of A (and they continue to exist after the move). These two moves may be depicted as 

and 

, with the initial position of the move at the bottom, and its final position at the top. When a move is played by a player, the rest of the position is left untouched. For example, 

is a move on a position with two players, with the left player playing and the right player left untouched. A *play* is then just a vertical pasting of moves, up to permutation of independent moves. A morphism of plays $f : P \to Q$ is an injective mapping of moves of P to those of Q that respect the play structure: $f(x)$ must have the same type as x, and if the player who plays x is equal to the one who plays y, then the players who play $f(x)$ and $f(y)$ must be equal too (and similarly if x creates a player who is equal to the one who plays y, etc).

As a first step (step 1 in Figure 1), we map our example play s to its P-view tree [3], a tree whose branches are the HON-views of s. There are two types of "arrows" in this tree: the ones depicted as real arrows, which correspond exactly to the ones in s, and the ones that create the tree structure, which correspond to the *view* of each move. This tree has some nice properties: in particular, it may only branch at odd depth. We then map this tree (step 2) to a proof tree in a sequent calculus based on arenas. A node x labelled by a sequent S in this tree may be a child of y labelled by S' exactly when, in our game, a player labelled by S' may make a move that produces of player labelled by S. Just like P-view trees, these proof trees may branch with an arbitrary arity, but only at odd depth. Even though both notions are trees and look alike (the structure is clearly similar), there are some subtle differences: the P-view tree is labelled by moves, while the proof tree is labelled by sequents of arenas; the P-view tree contains pointers, while the proof tree does not. A little bit of work has to be done to prove that this mapping is a full embedding. We give a proof for the composite of steps 1 and 2 in Section 4.

Steps 3 and 4 are conceptually simple, but for lack of space, we do not explain them in the rest of the paper. Step 3 maps the proof tree, which may branch with an arbitrary degree, to a similar tree that only branches in a binary way. Let us call the result of step 3 a *sequential* proof tree (in the sense that we have sequentialised some of the tree structure). To do this, we change the rules on which our proof trees are built. Here, the rules that have a positive sequent as conclusion are the same as in the previous proof trees, but the rules that have a negative sequent as conclusion have necessarily two premises: a positive one like in the previous proof trees, and a copy of the conclusion. Sequentialising a proof tree is now completely obvious: if a node has n premises, we just apply the sequential rule

n times. There is a choice to be made on the order in which we sequentialise the rules, but all these choices lead to isomorphic plays in the end. For example, we have decided to linearise the tree by applying t_l first, and then f_l in our example. Step 4 is completely direct: all the rules of the proof tree now exactly correspond to moves in our game, so we just mimic the structure of the proof tree. Nodes in the tree become players with the same label, and deduction rules become moves in our game. Here again, since the tree is a branching structure, we may choose to apply the moves in different orders, but all choices lead to isomorphic plays. In our example, we have chosen to play t_l, f_r, f_l, and t_r in that order, but we could have played f_r at any point after t_l (and all the resulting plays are isomorphic).

To show that this mapping reduces to an equivalence when restricted to views on both sides, we must explain what a view is in our setting. If we see the equal signs of a play as mathematical equality (which is indeed the case in the actual definition of plays), then we may collapse the whole structure along these equal signs to obtain a tree structure. Let us call such a structure a *play tree*. A play is a view when its play tree is non-branching. Now, our embedding reduces to an equivalence when restricted to views because HON-views are mapped to non-branching P-view trees, which are ultimately mapped to non-branching plays, i.e., views, and views conversely all come from non-branching proof trees, which all come from non-branching P-view trees, which in turn all come from HON-views.

Let us finally mention the case of morphisms. Morphisms of HON-plays are injective functions that respect views. Through step 1, they are exactly transported to (injective) morphisms of trees. Through step 2, they are exactly transported to inclusions of proof trees. Now, the easiest way to understand the equivalence, in terms of morphisms, is to see the resulting play as a tree by collapsing all equal signs, as explained above. Then inclusions of proof trees are exactly transported to inclusions of play trees, which are exactly morphisms of plays.

We thus obtain, for each pair of arenas A and B, a commuting square

$$\begin{array}{ccc} \mathbb{V}_{A,B} & \xhookrightarrow{\mathsf{i}_{HON}} & \mathbb{P}_{A,B} \\ {\scriptstyle F^{\mathbb{V}}}\downarrow & & \downarrow{\scriptstyle F} \\ \mathbb{E}^{\mathbb{V}}(A \vdash B) & \xhookrightarrow[\mathsf{i}]{} & \mathbb{E}(A \vdash B) \end{array} \tag{1}$$

of embeddings of categories, where i_{HON} denotes the embedding of HON-views into HON-plays, i denotes the embedding of views into plays, and $F^{\mathbb{V}}$ and F denote the constructed embeddings respectively from HON-views into views and from HON-plays into plays. Our first result is that all these embeddings are full and that $F^{\mathbb{V}}$ is an equivalence of categories (Theorem 29).

1.3 The level of strategies

However, we are not only interested in comparing views and plays, but also *innocent strategies*, which are at the core of game semantics. The square (1) gives a correspondence at the level of plays, but it also yields a tight correspondence between strategies in both approaches. More precisely, it induces an equivalence between innocent strategies in both contexts and shows that this equivalence is compatible with *innocentisation*.

To be more precise, there are two notions of innocent strategy in standard game semantics: the first one is a prefix-closed set of views, the second one is a prefix-closed set of plays verifying an extra condition called *innocence*. In both approaches, the first notion generalises to *presheaves* on views, which we call *behaviours* and *TO-behaviours* (for

Tsukada-Ong). The second notion generalises to *sheaves* on plays, which we simply call *innocent strategies* and *innocent TO-strategies*. In particular, mere presheaves on plays are possibly non-innocent strategies.

The idea behind this generalisation is the following: a prefix-closed set of views (in, say, $\mathbb{V}_{A,B}$) is a presheaf of booleans $B\colon\mathbb{V}_{A,B}^{op}\to 2$ (where 2 is the ordinal $0\to 1$ viewed as a category). Similarly, a prefix-closed set of plays is a presheaf $S\colon\mathbb{P}_{A,B}^{op}\to 2$. This presheaf is a sheaf for the Grothendieck topology induced by the embedding of $\mathbb{V}_{A,B}$ into $\mathbb{P}_{A,B}$ when $S(p)$ is accepted if and only if $S(v)$ is accepted for all views $v\to p$ (which exactly states *innocence*, the condition stating that a player can change its behaviour only according to what they have "seen", i.e., their view, of the play).

However, that is not sufficient to model concurrent strategies, because there may be several different ways to accept a play (or, in other words, a machine may be in several different states after a given trace). The classical example is that of Milner's coffee machines, which are the labelled transition systems depicted below.

Both machines accept exactly the same traces: ε, a, ab, and ac. However, one has a single way of accepting the trace a, after which it still accepts b and c, while the other makes a choice when accepting a whether to accept b or c. The first machine has one way of accepting a, while the second has two. Thus, modelling concurrent strategies correctly requires to know all the different states the strategy can end in after reading a trace: it is not a presheaf of booleans, but a presheaf of sets.

The functors F and $F^{\mathbb{V}}$ give rise to $\Delta_F\colon\mathbb{E}(\widehat{A\vdash B})\to\widehat{\mathbb{P}_{A,B}}$ and $\Delta_{F^{\mathbb{V}}}\colon\mathbb{E}^{\mathbb{V}}(\widehat{A\vdash B})\to\widehat{\mathbb{V}_{A,B}}$, where Δ_f is precomposition by f^{op}. Since $F^{\mathbb{V}}$ is an equivalence of categories, so is $\Delta_{F^{\mathbb{V}}}$, so behaviours and TO-behaviours are equivalent.

A strategy is *innocent* when it is in the essential image of $\prod_{\mathsf{i}}$ (or $\prod_{\mathsf{i}_{HON}}$), where $\prod_f$ denotes right Kan extension along f^{op}. This may also be seen as a sheaf condition stating that an innocent strategy accepts a play if and only if it accepts all the views that can be embedded into that play. Using Guitart's theory of *exact squares* [8], the square (1) provides a categorical explanation of why both induced categories of innocent strategies are equivalent. Indeed, it is exact (Corollary 33), which means that the square

$$\begin{array}{ccc}\widehat{\mathbb{V}_{A,B}} & \xhookrightarrow{\prod_{\mathsf{i}_{HON}}} & \widehat{\mathbb{P}_{A,B}}\\ {\scriptstyle\Delta_{F^{\mathbb{V}}}}\uparrow & & \uparrow{\scriptstyle\Delta_F}\\ \mathbb{E}^{\mathbb{V}}(\widehat{A\vdash B}) & \xhookrightarrow[\prod_{\mathsf{i}}]{} & \mathbb{E}(\widehat{A\vdash B})\end{array}\tag{2}$$

commutes up to isomorphism. In other words, Δ_F turns into an equivalence when restricted to innocent strategies and the *innocentisation* functors $\prod_{\mathsf{i}}$ and $\prod_{\mathsf{i}_{HON}}$ (which map any behaviour to the innocent strategy that "behaves similarly") are compatible with this equivalence.

▶ Remark. Had we wanted to make F an equivalence of categories rather than a mere full embedding, we could easily have imposed an additional condition on our plays akin to alternation in a classical HON-game setting. The point is that we want to compare the purely diagrammatic notion of play with the classical one. And since we obtain an

equivalence between both notions of innocent strategies anyway, we feel the result is in fact more convincing.

Note however that this work does not take composition of strategies into account, which is admittedly the Achilles' heel of the string diagrammatic approach as it stands now. Indeed, while Tsukada and Ong prove that their innocent strategies compose, we still don't know how to compose strategies into cut-free strategies in our setting. (The equivalence between both notions of innocent strategies actually gives a way to compose innocent strategies in our setting by composing in the other model, but it is not exactly an illuminating result, and in particular does not lift to non-innocent strategies.)

1.4 Related work

This paper compares Tsukada and Ong's model [19] to a model inspired by *playgrounds* [9, 10, 6] and which builds on ideas from *presheaf models* [12], *causal models* [17], and *game semantics* [1]. The notion of play, which is based on string diagrams, is close in spirit to Melliès's work [15]. The notion of trees that we use to bridge the gap between both models is reminiscent of the notion of legally justified trees (also known as P-view trees) by Boudes [3]. Let us also mention different approaches to concurrent game semantics [7, 13], based on variations of traditional games semantics, or [18], based on event structures.

Another paper by Tsukada and Ong [20] is quite close to this work. Their main result is a link between the notion of HON-play and another well-known notion of terms. The papers differ in that Tsukada and Ong link the notion of HON-play to notions that do not belong to game semantics (namely *resource terms*), thus finding new links between game semantics and other models, while we want to connect different models of game semantics.

The reader will notice that the interpretation of terms as strategies is not treated in the current paper. This point is treated in an unpublished paper [5], in which we use singular and geometric realisation functors to give an interpretation of terms of the non-deterministic λ-calculus studied by Tsukada and Ong into innocent strategies, show that it is the same as Tsukada and Ong's, and recover the *definability* result (that any innocent strategy is isomorphic to the interpretation of a normal form).

1.5 Overview

We start by recalling standard notions of game semantics as well as Tsukada and Ong's work in Section 2. Then, in Section 3, we set out to define two new models of HON games, one based on string diagrams, the other on proof trees, and show that the two have equivalent categories of plays, views, and strategies. In Section 4, we use the equivalences proved in the previous section to give the core result of this paper, which is the relationship between the categories of plays and views in our model based on string diagrams and Tsukada and Ong's. Finally, in Section 5, we build on the relationships shown in Section 4 to show the second result of this paper, which is that both models have equivalent categories of strategies, and that this equivalence is compatible with the embedding of behaviours into strategies.

2 Tsukada and Ong's model

Let us start with a brief recapitulation on Tsukada and Ong's categories of views and plays, as well as their notion of strategy.

As usual in game semantics, games are based on arenas (Definition 1).

▶ **Notation 3.** *We denote by $\sqrt{A}$ the set of roots of A. If A is an arena and m is in $\sqrt{A}$, then $A \cdot m$ is the arena strictly below m. If A is an arena, we denote by $m.A$ the tree t whose root is m and such that $t \cdot m = A$. Note that any arena is a coproduct of trees, so it can be written $A = \sum_{m \in \sqrt{A}} m.(A \cdot m)$.*

Let us fix arenas A and B. Let $A \rightarrowtriangle B$ denote the simple graph obtained by adding to $A + B$ an edge $b \to a$ for all $b \in \sqrt{B}$ and $a \in \sqrt{A}$ (if B is non-empty, otherwise $A \rightarrowtriangle B = \varnothing$). The notion of ownership straightforwardly extends to $A \rightarrowtriangle B$ since all paths from any root to some vertex v have the same length. Concretely, ownership is left unchanged in B but reversed in A.

▶ **Definition 4.** A *justified sequence* on (A, B) consists of a natural number $n \in \mathbb{N}$, equipped with maps $f : n \to M_A + M_B$ and $\varphi : n \to \{0\} \uplus n$ (here and later in the paper, we use n to denote the set $\{1, \dots, n\}$) such that, for all $i \in n$,

- $\varphi(i) < i$,
- if $\varphi(i) = 0$ then $f(i) \in \sqrt{B}$, and
- if $\varphi(i) \neq 0$, then $f(\varphi(i))$ is a parent of $f(i)$ in $A \rightarrowtriangle B$.

We will draw a justified sequence (n, f, φ) as the sequence of its $f(i)$'s, with arrows to denote φ, as is standard in game semantics.

For any $i \in n$, the *view* $\lceil (n, f, \varphi) \rceil_i$ *of* i *in* (n, f, φ) is the subset of n defined inductively by:

- $\lceil (n, f, \varphi) \rceil_i = \{i\}$ if i is an Opponent move with $\varphi(i) = 0$,
- $\lceil (n, f, \varphi) \rceil_i = \lceil (n, f, \varphi) \rceil_j \cup \{i\}$ if i is an Opponent move with $\varphi(i) = j > 0$,
- $\lceil (n, f, \varphi) \rceil_i = \lceil (n, f, \varphi) \rceil_{i-1} \cup \{i\}$ if i is a Proponent move.

A justified sequence $s = (n, f, \varphi)$ on (A, B) is *P-visible* when, for all Proponent moves i, $\varphi(i) \in \lceil s \rceil_i$. We further say that s is *alternating* when, for all $i \in n-1$, $\lambda_{A \rightarrowtriangle B}(i) \neq \lambda_{A \rightarrowtriangle B}(i+1)$.

▶ **Definition 5.** A *HON-play* on the pair of arenas (A, B) is a P-visible, alternating, justified sequence on (A, B) of even length.

A morphism of HON-plays $g : (n, f, \varphi) \to (n', f', \varphi')$ is an injective map $g : n \to n'$ such that: $f'(g(i)) = f(i)$ for all $i \in n$, $\varphi'(g(i)) = g(\varphi(i))$ for all $i \in n$ (with the convention that $g(0) = 0$), and $g(2i) = g(2i - 1) + 1$ for all $i \in n/2$. The last condition states that g should preserve blocks of an Opponent move and the next Proponent move (so-called *OP-blocks*).

▶ **Proposition 6.** *HON-plays and morphisms between them form a category $\mathbb{P}_{A,B}$, with composition given by composition of underlying maps.*

▶ **Example 7.** The sequence at the top-left of Figure 1 is a play on $(\mathbb{B}, \mathbb{B})$, where we have written m_l when m is played in the left-hand copy of $\mathbb{B}$, and m_r when it is played in the right-hand one.

▶ **Definition 8.** If $s = (n, f, \varphi)$ is a justified sequence, i and j are in n, and $\varphi(j) = 0$, we say that i is *hereditarily justified* by j if $i = j$ or $\varphi(i)$ is hereditarily justified by j.

A *thread* of s is a maximal sub-sequence of s in which all moves have the same hereditary justifier. The pointers of a thread are inherited from s.

▶ **Definition 9.** A *HON-view* on (A, B) is a non-empty HON-play $s = (n, f, \varphi)$ such that $\lceil s \rceil_n = s$. Let $\mathbb{V}_{A,B}$ denote the full subcategory of $\mathbb{P}_{A,B}$ spanning HON-views.

▶ **Proposition 10.** *A HON-play $s = (n, f, \varphi)$ is a HON-view if and only if for all odd $i \in n$, $\varphi(i) = i - 1$.*

The embedding $\mathsf{i}_{HON} : \mathbb{V}_{A,B} \hookrightarrow \mathbb{P}_{A,B}$ induces an adjunction $\widehat{\mathbb{P}_{A,B}} \underset{\Pi_{\mathsf{i}_{HON}}}{\overset{\Delta_{\mathsf{i}_{HON}}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \widehat{\mathbb{V}_{A,B}}$.

▶ **Definition 11.** We call $\widehat{\mathbb{V}_{A,B}}$ the category of *TO-behaviours*. We denote by $\mathrm{Sh}(\mathbb{P}_{A,B})$ [1] the category of *innocent TO-strategies* on (A, B), which is the essential image of $\prod_{\mathsf{i}_{HON}}$.

By full faithfulness of i_{HON}, $\prod_{\mathsf{i}_{HON}}$ restricts to an equivalence $\mathrm{Sh}(\mathbb{P}_{A,B}) \simeq \widehat{\mathbb{V}_{A,B}}$.

3 Two concurrent models of HON games

In this section, we build two models of HON games, one based on string diagrams, the other on proof trees, and show that they have equivalent categories of plays, views, and strategies.

3.1 String diagrams

The first model we build is in the same spirit as our previous models of CCS and the π-calculus. There are several salient differences between the notions of views and plays in our model and those from standard game semantics: first, they are interpreted *causally*, when the standard ones are interpreted *temporally* (i.e., our notion of play only retains causal dependency between moves, and there is no fixed order between moves that are independent from one another); secondly, they are intrinsically *multi-party* and put a strong focus on the notion of *position*, while standard HON games are two-player games, and the theory is devoid of the notion of position (though it exists in other settings [2]).

Let us first give an intuition of how to build models in this setting. The idea is to start from a sequent calculus that is an operational description of the calculus we study, and to build a multi-player game from it. Sequents show what positions should be, and derivation rules show what moves players are allowed to play. Plays are then simply sequences of moves, up to permutation of independent moves. While this sounds like it involves a cumbersome quotient of sequences of moves, the formal definition based on presheaves does not involve any such quotient. We will later design another model based on proof trees for a sequent calculus, but the the sequent calculus we introduce right now is just a tool internal to our games and has nothing to do with proof trees. In our case, the sequent calculus that will guide our construction is:

$$\frac{}{\Gamma, A \cdot m \vdash}\,{}^{\Lambda_{(\Gamma \vdash A), m}} \qquad \frac{}{\Gamma, A, \Delta \vdash A \cdot m}\,{}^{@_{(\Gamma, A, \Delta \vdash), |\Gamma|+1, m}} \qquad \frac{\Gamma \vdash A\Delta, A, \Delta' \vdash}{\Delta, \Gamma, \Delta' \vdash ,}\,{}^{\text{CUT}} \tag{3}$$

where sequents are lists of arenas, with possibly a distinguished arena, written $(A_1, \ldots, A_n \vdash)$ or $(A_1, \ldots, A_n \vdash A)$. Let Γ range over lists of arenas, $|\Gamma|$ denote the length of Γ, and for all $i \in |\Gamma|$, Γ_i the ith arena of Γ.

[1] As the notation $\mathrm{Sh}(-)$ suggests, this is also a category of *sheaves* for the Grothendieck topology embedding $\mathbb{V}_{A,B}$ into $\mathbb{P}_{A,B}$, though this fact is not used in this paper.

▶ Remark. The sequent calculus we use here does not make much sense from a logical point of view. It however makes sense when seen in relationship with the following sequent calculus:

$$\text{RIGHT}\ \frac{\ldots \quad \Gamma, A\cdot m \vdash \quad \ldots \quad (\forall m \in \sqrt{A})}{\Gamma \vdash A} \qquad \text{LEFT}\ \frac{\Gamma, A, \Delta \vdash A\cdot m}{\Gamma, A, \Delta \vdash} \qquad \text{CUT}\ \frac{\Gamma \vdash A\Delta, A, \Delta' \vdash}{\Delta, \Gamma, \Delta' \vdash .} \tag{4}$$

Note that this sequent calculus is a fragment of intuitionistic logic when an arena $A = \sum_{i\in n} m_i.A_i$ is interpreted as $[\![A]\!] = \bigwedge_{i\in n} (\neg\, [\![A_i]\!])$. Proofs in this calculus therefore correspond to proofs of intuitionistic logic. The objects of interest in calculus (3) are partial proofs (proofs whose branches may be left unfinished), which correspond to *explorations* of proofs in calculus (4).

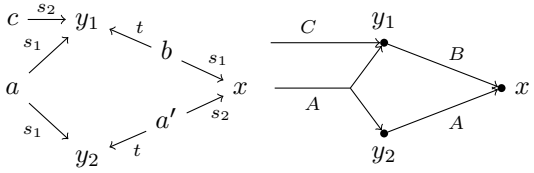
The notion of position we have given in the introduction is a simplification of the real notion of position. In our approach, positions are some kind of graphs. We call their vertices *players* and their edges *channels*. The idea is that players correspond to placeholders for program fragments, and thus for proofs of a certain type, while channels are the way players use to communicate with other players during a play. In the case of HON games, our game is based on arenas, and so is our sequent calculus. Positions will thus be some kind of graphs, whose vertices are labelled by such sequents, and whose edges are labelled by arenas. Players labelled $(\Gamma \vdash A)$ are linked to $|\Gamma|$ incoming channels of type Γ_i and to one outgoing channel of type A, and similarly for players labelled $(\Gamma \vdash)$. The discussion at the end of Example 13 gives more intuition on this notion. Positions are formally represented as presheaves over the following base category:

▶ **Definition 12.** Let $\mathbb{L}_1$ be the category with objects all arenas and sequents, and morphisms $s_i{:}\Gamma_i \to (\Gamma \vdash)$, $s_i{:}\Gamma_i \to (\Gamma \vdash A)$, and $t{:}A \to (\Gamma \vdash A)$.

An object of $\widehat{\mathbb{L}_1}$ is exactly a set of channels labelled A for each arena A, as well as a set of players labelled S for each sequent S, and maps mapping players to their incoming and outgoing channels. A natural transformation $X \to Y$ between such presheaves is akin to a graph morphism, in the sense that it sends each player labelled S in X to some player labelled S in Y (and similarly for channels), while preserving incoming and outgoing channels.

▶ **Example 13.** It is often helpful to represent positions (and more generally plays) graphically. We here give an example of a position given by a presheaf X, which we describe in mathematical terms on the left below (we only give the different sets that compose the presheaf, the functions can be inferred from the category of elements). We also draw its category of elements to the side, and finally how we draw this particular position. Note that the drawing of the position reflects exactly the structure of the presheaf, since it is just another representation of the category of elements (except for the order of the elements of the lists composing the sequents).

- $X(A) = \{a, a'\}$, $X(B) = \{b\}$, $X(C) = \{c\}$,
- $X(A, C \vdash B) = \{y_1\}$, $X(A \vdash A) = \{y_2\}$, $X(B, A \vdash) = \{x\}$,
- $X(-)$ empty otherwise.



In this particular position, there are three players x, y_1 and y_2, and four channels on which they may communicate. There is no need to depict positive and negative players differently

anymore: negative players have an outgoing channel, while positive ones don't. We see two interesting phenomena on this position: there is an input channel shared between y_1 and y_2, and some channels have an open end. In terms of semantics, the first phenomenon corresponds to the fact that several players may use a given proof of some formula A, while the second one corresponds to the fact that some resources may be given by the environment, instead of another player.

We now want to express the dynamics of the game. We start by an informal description in terms of a sequent calculus before augmenting $\mathbb{L}_1$ into a category $\mathbb{L}$ that will represent this dynamics. Our sequent calculus again guides us to desing this dynamics. The CUT rule shows when two players may interact: when the first player's outgoing channel is equal to one of the second player's incoming channels. The interaction between players is governed by the shapes of the Λ and @ rules. In other words, two players can interact according to the Λ and @ rules if and only if they are linked by a CUT rule. Now we just have to show the "shape" of this interaction.

As prescribed by Curry-Howard, we see this interaction as a step in a form of proof of a certain formula. We write $A = \sum_{i \in n} m_i.A_i$ for the arena shared by the two players. Let x by the negative player, labelled $(\Gamma \vdash A)$, and y the positive one, labelled $(\Delta, A, \Delta' \vdash)$. x should provide a proof of $[\![A]\!]$, while y should require one to prove a contradiction. When they interact, y chooses some $i \in n$ and asks x for a proof of $\neg [\![A_i]\!]$, i.e., y now provides a proof of $[\![A_i]\!]$ which x may inspect to prove a contradiction. Therefore, y turns into $(\Delta, A, \Delta' \vdash A_i)$, while x turns into $(\Gamma, A_i \vdash)$.

Moreover, x should not only change into a proof of contradiction assuming $[\![A_i]\!]$, but since y may inspect the proof of $[\![A]\!]$ again, the original x player should also remain in the final position.

In order to model those moves, we augment the base category $\mathbb{L}_1$:

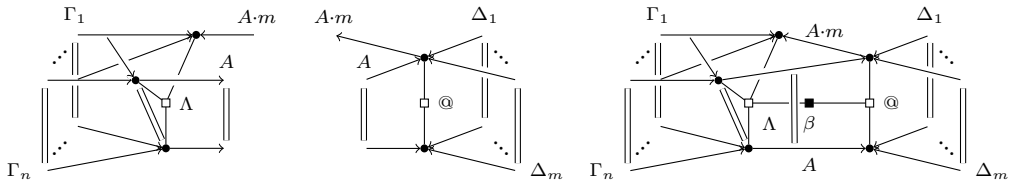
▶ **Definition 14.** Let $\mathbb{L}$ be the category $\mathbb{L}_1$ augmented with an object named after all Λ and @ rules in (3), as well as an object $\beta_{(\Gamma \vdash A),(\Delta \vdash),i,m}$ for all Γ, A, Δ, $i \in |\Delta|$ such that $\Delta_i = A$, and $m \in \sqrt{A}$; and with morphisms $t : S \to R$ for each sequent S that is the conclusion of a rule R, and $s : S \to R$ for each sequent S that is the premise of a rule R, as well as two morphisms $\lambda : \Lambda_{S,m} \to \beta_{S,S',i,m}$ and $@ : @_{S',i,m} \to \beta_{S,S',i,m}$ for all S, S', i, and m. These morphisms are quotiented by:

$$
\begin{array}{ll}
t \cdot s_i = s \cdot s_i & \text{for all rules } R \text{ and } i \text{ such that } s_i \text{ is defined,} \\
\lambda \cdot t \cdot t = @ \cdot t \cdot s_i & \text{for all } \beta_{S,S',i,m}, \\
\lambda \cdot s \cdot s_{|\Gamma|+1} = @ \cdot s \cdot t & \text{for all } \beta_{(\Gamma \vdash A),S',i,m}.
\end{array}
$$

In simple terms, we add an object for each possible basic shape of move in the game: objects Λ and @ for the rules they respectively represent, and objects β for each possible interaction of two rules.

This is however not sufficient to model moves. Indeed, moves have initial and final positions. To take this fact into account, we model moves not only by a representable, but by a cospan $Y \to \mathsf{y}_m \leftarrow X$, where X and Y are positions (i.e., presheaves empty except over $\mathbb{L}_1$), with X the initial position of m and Y its final position.

▶ **Definition 15.** The *seeds* corresponding to the representables $\Lambda_{(\Gamma \vdash A),m}$, $@_{(\Delta \vdash),i,m}$, and $\beta_{(\Gamma \vdash A),(\Delta \vdash),i,m}$ are the cospans represented graphically below.



Each drawing is the representation of the category of elements of the corresponding representable, with its initial position at the bottom and its final one on top, and where the squares correspond to the elements of type Λ, @, and β respectively (see the long version of this article [4] for formal definitions).

Apart from the existence of channels, there is a single difference between the moves we have shown in the introduction and the moves shown above: the β moves, which correspond to synchronisation. However, it can be shown that β moves can never be played in plays over $(A \vdash B)$, so we may treat them as non-existent. We still show them because they arise naturally in our game and they are necessary to model plays on more complex positions.

Once we have defined seeds, we reuse the same machinery as for CCS and the π-calculus to produce notions of plays and views, which we organise into categories. Since the description is long and technical, and of interest only to some readers, we here content with the following informal sketch. In seeds, all of the position participates in the move: a *move* is any cospan obtained by gluing (by pushout in $\widehat{\mathbb{L}}$) some seed and some position. In any categories with pushouts, cospans form the morphisms of a well-known bicategory. A *play* is any cospan isomorphic to some finite composite of moves in that bicategory.

▶ **Example 16.** Step 4 in Figure 1 produces a composite of six moves, whose gluing does not enforce any particular ordering between f_r and, say f_l.

Plays over any position X are the objects of a category $\mathbb{E}(X)$, whose morphisms are injective morphisms between underlying presheaves, preserving the initial position. A *view* is then merely a "non-branching" play of positive, even length. Views over X form a full subcategory $\mathbb{E}^{\mathbb{V}}(X)$ of $\mathbb{E}(X)$.

▶ **Lemma 17.** *For any sequent S, $\mathbb{E}(S)$ (resp. $\mathbb{E}^{\mathbb{V}}(S)$) is equivalent to a subcategory of $S/\widehat{\mathbb{L}}$ spanning morphisms $S \to U$ such that there exists a play (resp. view) $Y \to U \leftarrow X$.*

In order to define the different notions of strategies and link them to Tsukada and Ong's work, we restrict our attention to positions containing a single player $(A \vdash B)$ and both its channels, for any pair of arenas (A, B).

▶ **Definition 18.** A *behaviour* is a presheaf over $\mathbb{E}^{\mathbb{V}}(A \vdash B)$. A *strategy* is a presheaf over $\mathbb{E}(A \vdash B)$. An *innocent strategy* is a strategy in the essential image of $\prod_{\mathsf{i}}$.

3.2 Proof trees

In the long version of this paper [4], there is a direct proof of the correspondence between the notions of play and view in both approaches. However, that proof is technical and hardly illuminating. We here present another, more intuitive proof based on a simpler model that is equivalent to the one based on string diagrams in the particular case of plays whose initial position consists of one player $(A \vdash B)$.

This new model is based on proof trees in the following, *ad hoc* sequent calculus:

$$\text{RIGHT}\ \frac{\ldots \quad \Gamma, A \cdot m(i) \vdash \quad \ldots \quad (\forall i \in n)}{\Gamma \vdash A} \qquad \text{LEFT}\ \frac{\Gamma, A, \Delta \vdash A \cdot m}{\Gamma, A, \Delta \vdash} \tag{5}$$

where the RIGHT rule can be applied with any $m{:}n \to \sqrt{A}$ for a positive n and the LEFT rule can be applied with any $m \in \sqrt{A}$. Notice how, apart from the absence of a CUT rule, the sequent calculus (4) we have introduced at the beginning differs only slightly from this one: there, in the RIGHT rule, m must always be a bijection. Moreover, while the objects of interest are "complete" trees in the first case, the objects of interest here are "incomplete" trees. This once again corresponds to the fact that we are interested in explorations of proofs, and not proofs themselves. Therefore, some branches of the proof may not be explored (hence the fact that m need not be surjective) and others may be explored more than once (hence m need not be injective).

An *S-tree* is a partial proof tree (i.e., a proof tree whose branches may be left unfinished) whose conclusion is S. We define $\mathbb{T}(S)$ to be the category of S-trees and morphisms of such, where morphisms of S-trees are inclusions, both in width and depth, of S-trees, and are defined inductively by:

- the empty S-tree has exactly one morphism to all other S-trees,
- the set of morphisms from $\dfrac{T_1 \quad \cdots \quad T_n}{\Gamma \vdash A}$ to $\dfrac{T'_1 \quad \cdots \quad T'_m}{\Gamma \vdash A}$ is the disjoint union, for all injective $g{:}n \to m$, of $\prod_{i \in n} \mathbb{T}(\Gamma, A \cdot m_i \vdash)(T_i, T'_{g(i)})$,
- the set of morphisms from $\dfrac{T}{\Gamma, A, \Delta \vdash}$ to $\dfrac{T'}{\Gamma, A, \Delta \vdash}$ is $\mathbb{T}(\Gamma, A, \Delta \vdash A \cdot m)(T, T')$.

Notice that morphisms treat the premises of a $(\Gamma \vdash A)$ node as if they were unordered.

▶ **Definition 19.** An *S-branch* is a non-branching S-tree (i.e., an S-tree whose RIGHT rules are all unary) of positive, even depth. Let $\mathbb{B}(S)$ be the full subcategory of $\mathbb{T}(S)$ spanning S-branches.

▶ **Example 20.** The tree at the top-right of Figure 1 is an example of $(\mathbb{B}, \mathbb{B})$-tree. Here is an example of $(A \vdash B)$-tree: $\dfrac{A, B \cdot m \vdash \qquad A, B \cdot m \vdash}{A \vdash B}\, m, m$. None of these trees are branches (they both branch at some point). Note that, while the first tree intuitively represents a HON-play, the other one does not, as it lacks alternation: it is as if Opponent had played m twice in a row. This example also shows that morphisms treat premises as if they were unordered, in the sense that there are two morphisms from that tree to itself (one that maps both branches to themselves and one that swaps them).

We will later need this technical result:

▶ **Proposition 21.** *$\mathbb{T}(A, B \vdash C)$ and $\mathbb{T}(A + B \vdash C)$ are isomorphic.*

As announced in the introduction:

▶ **Lemma 22.** *$\mathbb{T}(A \vdash B)$ and $\mathbb{E}(A \vdash B)$ are equivalent, and so are $\mathbb{B}(A \vdash B)$ and $\mathbb{E}^{\mathbb{V}}(A \vdash B)$, and these equivalences are such that the following square commutes.*

$$\begin{array}{ccc} \mathbb{B}(A \vdash B) & \longrightarrow & \mathbb{T}(A \vdash B) \\ \downarrow & & \downarrow \\ \mathbb{E}^{\mathbb{V}}(A \vdash B) & \longrightarrow & \mathbb{E}(A \vdash B) \end{array}$$

4 The level of views and plays

In the previous section, we have introduced the embedding $\mathbb{E}^{\mathbb{V}}(S) \to \mathbb{E}(S)$ of views into plays based on string diagrams, and shown it equivalent to the embedding $\mathbb{B}(S) \to \mathbb{T}(S)$ based on

partial proof trees. We now want to show in which sense the latter is related to Tsukada and Ong's embedding $\mathsf{i}_{HON}\colon \mathbb{V}_{A,B} \hookrightarrow \mathbb{P}_{A,B}$. Namely, we build a full embedding F from $\mathbb{P}_{A,B}$ to $\mathbb{T}(A \vdash B)$ and then show that it restricts to an equivalence $F^{\mathbb{V}}\colon \mathbb{V}_{A,B} \to \mathbb{B}(A \vdash B)$. This will then yield a further correspondence between categories of strategies.

Let us first build $F(s)$ by induction on s. If s is empty, $F(s)$ is simply the empty $(A \vdash B)$-tree. Otherwise, s can be written as a sum of threads (Definition 8) $s = \sum_{i \in n} t_i$. Now, each thread t_i is of the form $m_i^1 m_i^2 s_i$ for moves m_i^1, m_i^2 and a play s_i, and a slight generalisation of a result at the start of Section 3.5 in [19] gives:

▶ **Proposition 23.** *For all arenas A and B, if $\mathsf{T}_{A,B}$ is the full subcategory of $\mathbb{P}_{A,B}$ spanning threads, $\chi\colon (mm')/\mathsf{T}_{A,B} \to \mathbb{P}_{C,C\cdot m'}, (mm's) \mapsto s$ is an isomorphism, where $C = A + B \cdot m$.*

Each s_i can therefore be mapped to a $(C_i \vdash C_i \cdot m_i^2)$-tree recursively, where $C_i = A + B \cdot m_i^1$. By Proposition 21, this gives an $(A, B \cdot m_i^1 \vdash C_i \cdot m_i^2)$-tree $\widetilde{F}(s_i)$, which can in turn be composed with the LEFT and RIGHT rules to give $F(s)$ as below:

$$\dfrac{\ldots \qquad \dfrac{\widetilde{F}(s_i)}{A, B \cdot m_i^1 \vdash}\, m = m_i^2 \qquad \ldots}{A \vdash B.}\, m(i) = m_i^1$$

▶ **Lemma 24.** *F is a full embedding.*

Proof. Injectivity on objects and faithfulness are easy to prove. The proof of fullness is mostly straightforward, except that it requires a reading of the pointers of s inside $F(s)$ to show that the antecedent though F of a morphism of $(A \vdash B)$-trees is indeed a HON-morphism. We explain in details how to read pointers from $F(s)$ below. ◀

First, recovering the moves in s from $F(s)$ is easy: they are all the m's used in any LEFT or RIGHT rule, with the obvious multiplicity. Recovering pointers is a bit trickier and requires to know what an *occurrence* of an arena in a tree is, as well as what it means for a move to *create* such an occurrence, and what it means for a move to be *played on* the occurrence of an arena. Lemmas 142 and 148 from [4] then explain how to recover pointers from our structure of plays. They can be stated as follows: each move m is justified by the move that created the arena occurrence m was played on.

▶ **Definition 25.** Let T be an $(A \vdash B)$-tree. The set of *moves* occurring in T is the disjoint union of all the moves of the rules of T, where, in (3), there is a single move m occurring in the LEFT rule, and the moves that occur in the RIGHT rule are all the $m(i)$'s.

In the LEFT rule, *m plays on A* and *creates $A \cdot m$*. In the RIGHT rule, *$m(i)$ plays on A* and *creates $A \cdot m(i)$*. Additionally, if $\dfrac{\Gamma, A \cdot m(1) \vdash \quad \ldots \quad \Gamma, A \cdot m(n) \vdash}{\Gamma \vdash A}$ is the first rule of T, then $m(i)$ creates all the arenas in $\Gamma, A \cdot m(i) \vdash$.

The *occurrences* of an arena in an S-tree T is an arena that is either the negative arena in the conclusion of T (if it exists) or an arena created by a rule in T.

▶ **Example 26.** There are four occurrences of the empty arena $\varnothing$ in the tree of Figure 1, created by t_l, f_l, f_r, and t_r respectively. There are also two occurrences of the boolean arena: the first one exists ($\mathbb{B}_r$) at the beginning, and the second one ($\mathbb{B}_l$) is created by q_r (and shared by all sequents above in the tree). In the tree of Example 20, there are two occurrences of the A arena, created by both m moves. This is an artefact of playing on the arena pair (A, B) rather than on $A \to B$, but this is more natural in our setting (for example, the notion of *interaction sequence*, which is used to study composition of strategies, is the same as that of play, but on particular positions).

In the tree in Figure 1, the q_l move is played on the occurrence of $\mathbb{B}$ called $\mathbb{B}_l$, which is created by q_r, so q_l is justified by q_r. Similarly, f_r is played on the occurrence of $\{\mathsf{t},\mathsf{f}\}$ called $\{\mathsf{t},\mathsf{f}\}_r$, which is also created by q_r, so f_r is also justified by q_r. We thus recover the pointer structure from the P-view tree, i.e., the pointer structure of the play we started from.

▶ **Lemma 27.** *F restricts to a functor $F^{\mathbb{V}}\colon \mathbb{V}_{A,B} \to \mathbb{E}^{\mathbb{V}}(A \vdash B)$.*

Proof. Let us take a HON-view $s = (n, f, \varphi)$, and show that $F(s)$ is a branch. The proof trees of our sequent calculus can only branch with the use of a RIGHT rule. In that case, we get by the method described above to recover pointers that two Opponent moves are justified by the same Proponent move. But we know by Proposition 10 that all Opponent moves in s are justified by the preceding move, so there can be no two Opponent moves justified by the same Proponent move. ◀

▶ **Lemma 28.** *$F^{\mathbb{V}}$ is an equivalence of categories.*

Proof. Since i, i_{HON}, and F are fully faithful, $F^{\mathbb{V}}$ is also fully faithful by left cancellation. Now, to show that $F^{\mathbb{V}}$ is essentially surjective on objects, we simply need to build an antecedent through $F^{\mathbb{V}}$ of any view v. The candidate HON-view is given by taking all moves of v from the root to the top (this is unambiguous since v is non-branching) and pointers given by the method described above. All that is left is to verify that the candidate HON-view is indeed a HON-view, which is done by verifying that it is a HON-play and that all Opponent moves are justified by the preceding move. The first point is easy, since the way we have chosen the antecedent may be generalised to any play, and the antecedent verifies all properties of HON-plays, except perhaps for alternation and having even length, which are both trivial in our case because views do not branch and have even depth. The second point is obvious by construction. ◀

▶ Remark. The proof above gives some insight on the only fundamental difference between our plays and HON-plays: ours only verify a weak form of alternation, where Opponent may play several moves in a row, but Proponent may only answer once per Opponent move.

By putting everything together, we get:

▶ **Theorem 29.** *The square* (1) *commutes, F is a full embedding, and $F^{\mathbb{V}}$ is an equivalence of categories.*

5 The level of strategies

A useful tool to compare strategies and behaviours in both settings is Guitart's theory of *exact squares* [8], which we now recall.

▶ **Definition 30.** A *square* is a natural transformation as on the left of (6).

Any square yields by restriction a square as in the middle below, and so by adjunction a further square as on the right:

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle u}\downarrow & \overset{\phi}{\Longrightarrow} & \downarrow{\scriptstyle v} \\ C & \xrightarrow[g]{} & D \end{array} \qquad \begin{array}{ccc} \widehat{A} & \xleftarrow{\Delta_f} & \widehat{B} \\ {\scriptstyle \Delta_u}\uparrow & \overset{\Delta_\phi}{\Longleftarrow} & \uparrow{\scriptstyle \Delta_v} \\ \widehat{C} & \xleftarrow[\Delta_g]{} & \widehat{D} \end{array} \qquad \begin{array}{ccc} \widehat{A} & \xrightarrow{\Pi_f} & \widehat{B} \\ {\scriptstyle \Delta_u}\uparrow & \overset{\tilde{\phi}}{\Longleftarrow} & \uparrow{\scriptstyle \Delta_v} \\ \widehat{C} & \xrightarrow[\Pi_g]{} & \widehat{D}. \end{array} \tag{6}$$

▶ **Definition 31.** A square ϕ is *exact* when $\tilde{\phi}$ is an isomorphism.

The result we ultimately want to prove to show the tight relationship between both approaches is that the square (2) commutes up to isomorphism. This corollary reduces to exactness of (1) filled with the identity.

▶ **Lemma 32.** *Any square as on the left of* (6) *in which ϕ is an identity, u is an equivalence, and v is fully faithful is exact.*

▶ **Corollary 33.** *The square* (2) *commutes up to isomorphism.*

6 Conclusion

Even though Tsukada and Ong's approach differs significantly from ours in terms of presentation, both approaches define similar notions of play (even though our notion of play is slightly looser than the one from traditional game semantics) and views, and the resulting notions of behaviours and innocent strategies are related in a very strong way. This shows that the differences between the two approaches are mainly choices of presentation.

However, Tsukada and Ong's approach goes further than ours: they define a cartesian closed category of arenas and strategies, which allows them to compose strategies. Two steps are required to compose strategies, called parallel composition and hiding: the first executes two strategies in parallel, and the second one hides the middle arena. While parallel composition is easy to manipulate in our setting because our game is intrinsically multi-party, hiding could admittedly be more difficult to handle.

References

1 Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000. doi:10.1006/inco.2000.2930.

2 Samson Abramsky and Paul-André Melliès. Concurrent games and full completeness. In *LICS*, pages 431–442. IEEE, 1999. doi:10.1109/LICS.1999.782638.

3 Pierre Boudes. Thick subtrees, games and experiments. In *TLCA*, volume 5608 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2009. doi:10.1007/978-3-642-02273-9_7.

4 Clovis Eberhart and Tom Hirschowitz. Justified sequences in string diagrams: a comparison between two approaches to concurrent game semantics. Preprint, 2016. URL: https://hal.archives-ouvertes.fr/hal-01372582.

5 Clovis Eberhart and Tom Hirschowitz. Game semantics as a singular functor, and definability as geometric realisation. working paper or preprint, 2017. URL: https://hal.archives-ouvertes.fr/hal-01527171.

6 Clovis Eberhart, Tom Hirschowitz, and Thomas Seiller. An intensionally fully-abstract sheaf model for pi. In *CALCO*, volume 35 of *LIPIcs*, pages 86–100. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.CALCO.2015.86.

7 Dan R. Ghica and Andrzej S. Murawski. Angelic semantics of fine-grained concurrency. In *FoSSaCS*, volume 2987 of *Lecture Notes in Computer Science*, pages 211–225. Springer, 2004. doi:10.1007/978-3-540-24727-2_16.

8 René Guitart. Relations et carrés exacts. *Annales des Sciences Mathématiques du Québec*, 4(2):103–125, 1980.

9 Tom Hirschowitz. Full abstraction for fair testing in CCS. In *CALCO*, volume 8089 of *Lecture Notes in Computer Science*, pages 175–190. Springer, 2013. doi:10.1007/978-3-642-40206-7_14.

10 Tom Hirschowitz. Full abstraction for fair testing in CCS (expanded version). *Logical Methods in Computer Science*, 10(4), 2014. doi:10.2168/LMCS-10(4:2)2014.

11 J. M. E. Hyland and C.-H. Luke Ong. On full abstraction for PCF: I, II, and III. *Information and Computation*, 163(2):285–408, 2000. doi:10.1006/inco.2000.2917.

12 André Joyal, Mogens Nielsen, and Glynn Winskel. Bisimulation and open maps. In *LICS*, pages 418–427. IEEE, 1993. doi:10.1109/LICS.1993.287566.

13 James Laird. Game semantics for higher-order concurrency. In *FSTTCS*, volume 4337 of *Lecture Notes in Computer Science*, pages 417–428. Springer, 2006. doi:10.1007/11944836_38.

14 Paul-André Melliès. Asynchronous games 2: the true concurrency of innocence. In *Proc. 15th International Conference on Concurrency Theory*, volume 3170 of *Lecture Notes in Computer Science*, pages 448–465. Springer, 2004. doi:10.1007/978-3-540-28644-8_29.

15 Paul-André Melliès. Game semantics in string diagrams. In *LICS*, pages 481–490. IEEE, 2012. doi:10.1109/LICS.2012.58.

16 Hanno Nickau. Hereditarily sequential functionals. In *LFCS*, volume 813 of *Lecture Notes in Computer Science*, pages 253–264. Springer, 1994. doi:10.1007/3-540-58140-5_25.

17 M. Nielsen, G. Plotkin, and G. Winskel. Event structures and domains, part 1. *Theoretical Computer Science*, 13:65–108, 1981.

18 Silvain Rideau and Glynn Winskel. Concurrent strategies. In *LICS*, pages 409–418. IEEE, 2011. doi:10.1109/LICS.2011.13.

19 Takeshi Tsukada and C.-H. Luke Ong. Nondeterminism in game semantics via sheaves. In *LICS*. IEEE, 2015.

20 Takeshi Tsukada and C.-H. Luke Ong. Plays as resource terms via non-idempotent intersection types. In *LICS*, pages 237–246. ACM, 2016. doi:10.1145/2933575.2934553.

Disjunctive Bases: Normal Forms for Modal Logics*

Sebastian Enqvist[1] **and Yde Venema**[2]

1 **Department of Philosophy, Stockholm University, Sweden**
thesebastianenqvist@gmail.se
2 **Institute for Logic, Language and Computation, Universiteit van Amsterdam, Netherlands**
y.venema@uva.nl

Abstract

We present the concept of a disjunctive basis as a generic framework for normal forms in modal logic based on coalgebra. Disjunctive bases were defined in previous work on completeness for modal fixpoint logics, where they played a central role in the proof of a generic completeness theorem for coalgebraic mu-calculi. Believing the concept has a much wider significance, here we investigate it more thoroughly in its own right. We show that the presence of a disjunctive basis at the "one-step" level entails a number of good properties for a coalgebraic mu-calculus, in particular, a simulation theorem showing that every alternating automaton can be transformed into an equivalent nondeterministic one. Based on this, we prove a Lyndon theorem for the full fixpoint logic, its fixpoint-free fragment and its one-step fragment, and a Uniform Interpolation result, for both the full mu-calculus and its fixpoint-free fragment.

We also raise the questions, when a disjunctive basis exists, and how disjunctive bases are related to Moss' coalgebraic "nabla" modalities. Nabla formulas provide disjunctive bases for many coalgebraic modal logics, but there are cases where disjunctive bases give useful normal forms even when nabla formulas fail to do so, our prime example being graded modal logic.

Finally, we consider the problem of giving a category-theoretic formulation of disjunctive bases, and provide a partial solution.

1998 ACM Subject Classification I.2.4 Knowledge Representation Formalisms and Methods, F.1.1 Models of Computation, F.4.1 Mathematical Logic

Keywords and phrases Modal logic, fixpoint logic, automata, coalgebra, graded modal logic, Lyndon theorem, uniform interpolation

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.11

1 Introduction

The topic of this paper connects modal μ-calculi, coalgebra and automata. The connection between the modal μ-calculus, as introduced by Kozen [13], and automata running on infinite objects, is standard [9]. Many of the most fundamental results about the modal μ-calculus have been proved by making use of this connection, including completeness of Kozen's axiom system [23], and model theoretic results like expressive completeness [12], uniform interpolation and a Lyndon theorem [3].

* For a full version of this paper, containing proofs of all statements, see [6], http://www.illc.uva.nl/Research/Publications/Reports/PP-2017-05.text.pdf.

The standard modal μ-calculus was generalized to a generic, coalgebraic modal μ-calculi [21], of which the modal basis was provided by Moss' original coalgebraic modality [17], now known as the *nabla* modality. From a meta-logical perspective, Moss' nabla logics and their fixpoint extensions are wonderfully well-behaved. For example, a generic completeness theorem for nabla logics by a uniform system of axioms was established [14], and this was recently extended to the fixpoint extension of the finitary Moss logic [4]. Most importantly, the automata corresponding to the fixpoint extension of Moss' finitary nabla logic always enjoy a *simulation theorem*, allowing arbitrary coalgebraic automata to be simulated by *non-deterministic* ones; this goes back to the work of Janin & Walukiewicz on μ-automata [11]. The simulation theorem provides a very strong normal form for these logics, and plays an important role in the proofs of several results for coalgebraic fixpoint logics.

The downside of this approach is that the nabla modality is rather non-standard, and understanding what concrete formulas actually say is not always easy. For this reason, another approach to coalgebraic modal logic has become popular, based on so called *predicate liftings*. This approach, going back to the work of Pattinson [19], provides a much more familiar syntax in concrete applications, but can still be elegantly formulated at the level of generality and abstraction that makes the coalgebraic approach to modal logic attractive in the first place. (For a comparison between the two approaches, see [15].) Coalgebraic μ-calculi have also been developed as extensions of the predicate liftings based languages [2], and the resulting logics are very well behaved: for example, good complexity results were obtained in op. cit. Again, the connection between formulas and automata can be formulated in this setting [7], but a central piece is now missing: so far, no simulation theorem has been established for logics based on predicate liftings. In fact, it is not trivial even to define what a non-deterministic automaton *is* in this setting.

This problem turned up in recent work [5], by ourselves together with Seifan, where we extended our earlier completeness result for Moss-style fixpoint logics [4] to the predicate liftings setting. Our solution was to introduce the concept of a *disjunctive basis*, which formalizes in a compact way the minimal requirements that a collection of predicate liftings Λ must meet in order for the class of corresponding Λ-automata to admit a simulation theorem. Our aim in the present paper is to follow up on this conceptual contribution, which we believe is of much wider significance besides providing a tool to prove completeness results.

Exemplifying this, we shall explore some of the applications of our coalgebraic simulation theorem. Some of these transfer known results for nabla based fixpoint logics to the predicate liftings setting; for example, we show that a linear-size model property holds for our non-deterministic automata (or "disjunctive" automata as we will call them), following [21]. We also show that uniform interpolation results hold for coalgebraic fixpoint logics in the presence of a disjunctive basis, which was proved for the Moss-style languages in [16]. Finally, we prove a Lyndon theorem for coalgebraic fixpoint logics, generalizing a result for the standard modal μ-calculus proved in [3]: a formula is monotone in one of its variables if and only if it is equivalent to one in which the variable appears positively. We also prove an explicitly *one-step* version of this last result, which we believe has some practical interest for modal fixpoint logics: It is used to show that, given an expressively complete set of monotone predicate liftings, its associated μ-calculus has the same expressive power as the full μ-calculus based on the collection of all monotone predicate liftings.

Next to proving these results, we compare the notion of a disjunctive basis to the nabla based approach to coalgebraic fixpoint logics. The connection will be highlighted in Section 7 where we discuss disjunctive predicate liftings via the Yoneda lemma: here the Barr lifting of the ambient functor (on which the semantics of nabla modalities are based) comes into

the picture naturally. This is not to say that disjunctive bases are just "nablas in disguise": it is a fundamental concept, and in some cases it is the *right* concept as *opposed* to nabla formulas. As a clear example of this, we consider *graded modal logic*, which adds counting modalities to modal logic. While we will see that this language has a disjunctive basis, at the same time we will prove that no such basis can be based on the nabla modalities.

2 Preliminaries

We assume that the reader is familiar with coalgebra, coalgebraic modal logic and the basic theory of automata operating on infinite objects. The aim of this section is to fix some definitions and notations.

First of all, throughout this paper we will use the letter T to denote an arbitrary *set functor*, that is, a covariant endofunctor on the category Set having sets as objects and functions as arrows. For notational convenience we sometimes assume that T preserves inclusions; our arguments can easily be adapted to the more general case. Functors of coalgebraic interest include the identity functor Id, the *powerset functor* P, the monotone neighborhood functor M and the (finitary) bag functor B (where $\mathsf{B}S$ is the collection of *weight functions* $\sigma : S \to \omega$ with finite support). We also need the contravariant powerset functor $\breve{\mathsf{P}}$.

A T-*coalgebra* is a pair $\mathbb{S} = (S, \sigma)$ where S is a set of objects called *states* or *points* and $\sigma : S \to \mathsf{T}S$ is the *transition* or *coalgebra map* of $\mathbb{S}$. A *pointed* T-coalgebra is a pair $(\mathbb{S}, s)$ consisting of a T-coalgebra and a state $s \in S$. We call a function $f : S' \to S$ a *coalgebra homomorphism* from (S', σ') to (S, σ) if $\sigma \circ f = \mathsf{T}f \circ \sigma'$, and write $(\mathbb{S}', s') \rightrightarrows (\mathbb{S}, s)$ if there is such a coalgebra morphism mapping s' to s.

With $\mathtt{X}$ a set of proposition letters, a T-*model over* $\mathtt{X}$ is a pair $(\mathbb{S}, V)$ consisting of a T-coalgebra $\mathbb{S} = (S, \sigma)$ and a $\mathtt{X}$-*valuation* V on S, that is, a function $V : \mathtt{X} \to \mathsf{P}S$. The *marking* associated with V is the transpose map $V^\flat : S \to \mathsf{P}\mathtt{X}$ given by $V^\flat(s) := \{p \in \mathtt{X} \mid s \in V(p)\}$. Thus the pair $(\mathbb{S}, V)$ induces a $\mathsf{T}_\mathtt{X}$-coalgebra $(S, (V^\flat, \sigma))$, where $\mathsf{T}_\mathtt{X}$ is the set functor $\mathsf{P}\mathtt{X} \times \mathsf{T}$.

We will mainly follow the approach in coalgebraic modal logic where modalities are associated (or even identified) with finitary predicate liftings. A *predicate lifting* of arity n is a natural transformation $\lambda : \breve{\mathsf{P}}^n \Rightarrow \breve{\mathsf{P}}\mathsf{T}$. Such a predicate lifting is *monotone* if for every set S, the map $\lambda_S : (\mathsf{P}S)^n \to \mathsf{P}\mathsf{T}S$ preserves the subset order in each coordinate. The induced predicate lifting $\lambda^\partial : \mathsf{P}^n \Rightarrow \mathsf{P}\mathsf{T}$, given by $\lambda^\partial_S(X_1, \ldots, X_n) := \mathsf{T}S \setminus \lambda_S(S \setminus X_1, \ldots, S \setminus X_1)$, is called the *(Boolean) dual* of λ. A *monotone modal signature*, or briefly: *signature* for T is a set Λ of monotone predicate liftings for T, which is closed under taking boolean duals.

Given a signature Λ, the formulas of the *coalgebraic μ-calculus* $\mu\mathtt{ML}_\Lambda$ are given by the following grammar:

$$\varphi ::= p \mid \bot \mid \neg\varphi \mid \varphi_0 \vee \varphi_1 \mid \heartsuit_\lambda(\varphi_1, \ldots, \varphi_n) \mid \mu x.\varphi'$$

where p and x are propositional variables, $\lambda \in \Lambda$ has arity n, and the application of the fixpoint operator μx is under the proviso that all occurrences of x in φ' are positive (i.e., under an even number of negations). We let $\mathtt{ML}_\Lambda$ and $\mu\mathtt{ML}_\Lambda(\mathtt{X})$ denote, respectively, the fixpoint-free fragment of $\mu\mathtt{ML}_\Lambda$ and the set of $\mu\mathtt{ML}_\Lambda$-formulas taking free variables from $\mathtt{X}$.

Formulas of such coalgebraic μ-calculi are interpreted in coalgebraic models, as follows. Let $\mathbb{S} = (S, \sigma, V)$ be a T-model over a set $\mathtt{X}$ of proposition letters. By induction on the complexity of formulas, we define a *meaning function* $[\![\cdot]\!]^{\mathbb{S}} : \mu\mathtt{ML}_\Lambda(\mathtt{X}) \to \mathsf{P}S$, together with an associated *satisfaction relation* $\Vdash \subseteq S \times \mu\mathtt{ML}_\Lambda(\mathtt{X})$ given by $\mathbb{S}, s \Vdash \varphi$ iff $s \in [\![\varphi]\!]^{\mathbb{S}}$. All clauses of this definition are standard; for instance, the one for the modality $\heartsuit_\lambda$ is given by

$$\mathbb{S}, s \Vdash \heartsuit_\lambda(\varphi_1, \ldots, \varphi_n) \text{ if } \sigma(s) \in \lambda_S([\![\varphi_1]\!]^{\mathbb{S}}, \ldots, [\![\varphi_n]\!]^{\mathbb{S}}). \tag{1}$$

For the least fixpoint operator we apply the standard description of least fixpoints of monotone maps from the Knaster-Tarski theorem and take

$$[\![\mu x.\varphi]\!]^{\mathbb{S}} := \bigcap \{U \in \mathsf{P}S \mid [\![\varphi]\!]^{(S,\sigma,V[x\mapsto U])} \subseteq U\},$$

where $V[x \mapsto U]$ is given by $V[x \mapsto U](x) := U$ while $V[x \mapsto U](p) := V(p)$ for $p \neq x$. A formula φ is said to be *monotone* in a variable p if, for every T-model $\mathbb{S} = (S, \sigma, V)$ and all sets $Z_1 \subseteq Z_2 \subseteq S$, we have $[\![\varphi]\!]^{(S,\sigma,V[p\mapsto Z_1])} \subseteq [\![\varphi]\!]^{(S,\sigma,V[p\mapsto Z_2])}$.

Well-known examples of coalgebraic modalities include the next-time operator $\bigcirc$ of linear time temporal logic, the standard Kripkean modalities $\Box$ and $\Diamond$, the more general modalities of monotone modal logic, and the *counting modalities* $\Diamond^k$ and $\Box^k$ of graded modal logic, which can be interpreted over B-coalgebras using the predicate liftings $\underline{k}$ and $\overline{k}$ given by

$$\begin{aligned} \underline{k}_S :&\quad U \mapsto \{\sigma \in \mathsf{B}S \mid \textstyle\sum_{u\in U} \sigma(u) \geq k\} \\ \overline{k}_S :&\quad U \mapsto \{\sigma \in \mathsf{B}S \mid \textstyle\sum_{u\notin U} \sigma(u) < k\}. \end{aligned}$$

A pivotal role in our approach is filled by the one-step versions of coalgebraic logics. Given a signature Λ and a set A of variables, we define the set $\mathtt{Bool}(A)$ of *boolean formulas* over A and the set $\mathtt{1ML}_\Lambda(A)$ of *one-step* Λ*-formulas* over A, by the following grammars:

$$\begin{aligned} \mathtt{Bool}(A) \ni \pi &::= a \mid \bot \mid \top \mid \pi \vee \pi \mid \pi \wedge \pi \mid \neg\pi \\ \mathtt{1ML}_\Lambda(A) \ni \alpha &::= \heartsuit_\lambda \overline{\pi} \mid \bot \mid \top \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \neg\alpha \end{aligned}$$

where $a \in A$ and $\pi = (\pi_1, \ldots, \pi_n)$ for some $\lambda \in \Lambda$ of arity n. We will denote the positive (negation-free) fragments of $\mathtt{Bool}(A)$ and $\mathtt{1ML}_\Lambda(A)$ as, respectively, $\mathtt{Latt}(A)$ and $\mathtt{1ML}^+_\Lambda(A)$.

We shall often make use of substitutions: given a finite set A, let $\vee_A : \mathsf{P}A \to \mathtt{Bool}(A)$ be the map sending B to $\bigvee B$, and let $\wedge_A : \mathsf{P}A \to \mathtt{Bool}(A)$ be the map sending B to $\bigwedge B$, and given sets A, B let $\wedge_{A,B} : A \times B \to \mathtt{Bool}(A \cup B)$ be defined by mapping (a, b) to $a \wedge b$.

A monotone modal signature Λ for T is *expressively complete* if, for every n-place predicate lifting λ and variables $a_1, \ldots, a_n$ there is a formula $\alpha \in \mathtt{1ML}_\Lambda(\{a_1, \ldots, a_n\})$ which is equivalent to $\heartsuit_\lambda \overline{a}$. We will also be interested in the following strengthening of expressive completeness: we say that Λ is *Lyndon complete* if, for every *monotone* n-place predicate lifting λ and variables $a_1, \ldots, a_n$, there is a *positive* formula $\alpha \in \mathtt{1ML}^+_\Lambda(\{a_1, \ldots, a_n\})$ equivalent to $\heartsuit_\lambda \overline{a}$.

One-step formulas are naturally interpreted in the following structures. A *one-step* T*-frame* is a pair (S, σ) with $\sigma \in \mathsf{T}S$, i.e., an object in the category $\mathcal{E}(\mathsf{T})$ of elements of T. Similarly a *one-step* T*-model* over a set A of variables is a triple (S, σ, m) such that (S, σ) is a one-step T-frame and $m : S \to \mathsf{P}A$ is an A-marking on S. *Morphisms* of one-step frames and of one-step models are defined in the obvious way.

Given a one-step model (S, σ, m), we define the 0*-step interpretation* $[\![\pi]\!]^0_m \subseteq S$ of $\pi \in \mathtt{Bool}(A)$ by the obvious induction: $[\![a]\!]^0_m := \{v \in S \mid a \in m(v)\}$, $[\![\top]\!]^0_m := S$, $[\![\bot]\!]^0_m := \varnothing$, and the standard clauses for $\wedge, \vee$ and $\neg$. Similarly, the *one-step interpretation* $[\![\alpha]\!]^1_m$ of $\alpha \in \mathtt{1ML}_\Lambda(A)$ is defined as a subset of $\mathsf{T}S$, with $[\![\heartsuit_\lambda(\pi_1, \ldots, \pi_n)]\!]^1_m := \lambda_S([\![\pi_1]\!]^0_m, \ldots, [\![\pi_n]\!]^0_m)$, and standard clauses for $\bot, \top, \wedge, \vee$ and $\neg$. Given a one-step modal (S, σ, m), we write $S, \sigma, m \Vdash^1 \alpha$ for $\sigma \in [\![\alpha]\!]^1_m$. Notions like one-step satisfiability, validity and equivalence are defined in the obvious way.

A Λ*-automaton* over a set $\mathtt{X}$ of proposition letters, or more broadly, a *coalgebra automaton*, is a quadruple $\mathbb{A} = (A, \Theta, \Omega, a_I)$ where A is a finite set of *states*, with *initial state* $a_I \in A$, $\Theta : A \times \mathsf{P}\mathtt{X} \to \mathtt{1ML}^+_\Lambda(A)$ is the *transition map* and $\Omega : A \to \omega$ is the *priority map* of $\mathbb{A}$. Its semantics is given in terms of a two-player infinite parity game: With $\mathbb{S} = (S, \sigma, V)$ a T-model over a set $\mathtt{Y} \supseteq \mathtt{X}$, the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ is the parity game given by the table below.

Position	Player	Admissible moves	Priority
$(a, s) \in A \times S$	$\exists$	$\{m : S \to \mathsf{P}A \mid (S, \sigma(s), m) \Vdash^1 \Theta(a, \mathtt{X} \cap V^\flat(s))\}$	$\Omega(a)$
$m : S \to \mathsf{P}A$	$\forall$	$\{(b, t) \mid b \in m(t)\}$	0

We say that $\mathbb{A}$ *accepts* the pointed T-model $(\mathbb{S}, s)$, notation: $\mathbb{S}, s \Vdash \mathbb{A}$, if (a_I, s) is a winning position for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$.

▶ **Fact 1.** *There are effective constructions transforming a formula in $\mu\mathtt{ML}_\Lambda(\mathtt{X})$ into an equivalent Λ-automaton over $\mathtt{X}$, and vice versa.*

3 Disjunctive formulas and disjunctive bases

In this section, we present the main conceptual contribution of the paper, and define disjunctive bases. We then immediately consider a number of examples.

▶ **Definition 2.** A one-step formula $\alpha \in \mathtt{1ML}^+_\Lambda(A)$ is called *disjunctive* if for every one-step model (S, σ, m) such that $S, \sigma, m \Vdash^1 \alpha$ there is a one-step frame morphism $f : (S', \sigma') \to (S, \sigma)$ and a marking $m' : S' \to \mathsf{P}A$ such that:

1. $S', \sigma', m' \Vdash^1 \alpha$;
2. $m'(s') \subseteq m(f(s'))$, for all $s' \in S'$;
3. $|m'(s')| \leq 1$, for all $s' \in S'$.

▶ **Definition 3.** Let $\mathtt{D}$ be an assignment of a set of positive one-step formulas $\mathtt{D}(A) \subseteq \mathtt{1ML}^+_\Lambda(A)$ for all sets of variables A. Then $\mathtt{D}$ is called a *disjunctive basis* for Λ if each formula in $\mathtt{D}(A)$ is disjunctive, and the following conditions hold:

1. $\mathtt{D}(A)$ is closed under finite disjunctions (in particular, it contains $\top = \bigvee \varnothing$).
2. $\mathtt{D}$ is *distributive over* Λ: for every one-step formula of the form $\heartsuit_\lambda \overline{\pi}$ there is a formula $\delta \in \mathtt{D}(\mathsf{P}(A))$ such that $\heartsuit_\lambda \overline{\pi} \equiv^1 \delta[\wedge_A]$.
3. $\mathtt{D}$ *admits a binary distributive law*: for any two formulas $\alpha \in \mathtt{D}(A)$ and $\beta \in \mathtt{D}(B)$, there is a formula $\gamma \in \mathtt{D}(A \times B)$ such that $\alpha \wedge \beta \equiv^1 \gamma[\wedge_{A,B}]$.

Disjunctive bases for weak pullback preserving functors. It is not hard to prove that disjunctive formulas generalize the Moss modalities, which are tightly connected to weak pullback preservation of the coalgebraic type functor. (Due to space limitations we refer to [14] for the details on the syntax and semantics of the Moss modalities.) In many interesting cases this suffices to find a disjunctive basis.

▶ **Proposition 4.** *Let Λ be a signature for a weak-pullback preserving functor T. If Λ is Lyndon complete, then the collection of all (finite or infinite) disjunctions of nabla formulas provides a disjunctive basis for Λ.*

For a proof of this proposition, which is a fairly straightforward exercise in coalgebraic logic, we refer the reader to [6].

Graded modal logic. Our main motivating example to introduce disjunctive bases is graded modal logic. The bag functor does preserve weak pullbacks, and so its Moss modalities are disjunctive, and the set of all monotone liftings for B does admit a disjunctive basis as an instance of Proposition 4. Note, however, that this proposition does not apply to graded modal logic, since the signature Σ_B is not expressively complete; this was essentially shown in [18]. It was observed already in [1] that very simple formulas in the one-step language $\mathtt{1ML}_{\Sigma_\mathsf{B}}$ are impossible to express in the (finitary) Moss language; consequently, the Moss

modalities for the bag functor are not suitable to provide disjunctive normal forms for graded modal logic. Still, the signature Σ_{B} does have a disjunctive basis.

We say that a one-step model for the finite bag functor is *Kripkean* if all states have multiplicity 1. Note that a Kripkean one-step model (S, σ, m) can also be seen as a structure (in the sense of standard first-order model theory) for a first-order signature consisting of a monadic predicate for each $a \in A$: Simply consider the pair (S, V_m), where $V_m : A \to \mathsf{P}S$ is the interpretation given by putting $V_m(a) := \{s \in S \mid a \in m(s)\}$. We consider special basic formulas of monadic first-order logic of the form:

$$\gamma(\overline{a}, B) := \exists \overline{x}(\mathsf{diff}(\overline{x}) \wedge \bigwedge_{i \in I} a_i(x_i) \wedge \forall y(\mathsf{diff}(\overline{x}, y) \to \bigvee_{b \in B} b(y)))$$

It is not hard to see that any Kripkean one-step B-model (S, σ, m) satisfies:

$$S, \sigma, m \Vdash^1 \gamma(\overline{a}, B) \text{ implies } S, \sigma, m' \Vdash^1 \gamma(\overline{a}, B) \text{ for some } m' \subseteq m \text{ with } \mathsf{Ran}(m') \subseteq \mathsf{P}_{\leq 1}A. \quad (2)$$

We can turn the formula $\gamma(\overline{a}, B)$ into a modality $\nabla(\overline{a}; B)$ that can be interpreted in *all* one-step B-models, using the observation that every one-step B-frame (S, σ) has a unique Kripkean cover $(\widetilde{S}, \widetilde{\sigma})$ defined by putting $\widetilde{S} := \bigcup\{s \times \sigma(s) \mid s \in S\}$, and $\widetilde{\sigma}(s, i) := 1$ for all $s \in S$ and $i \in \sigma(s)$ (where we view each finite ordinal as the set of all smaller ordinals). Then we can define, for an arbitrary one-step B-model (S, σ)

$$S, \sigma, m \Vdash^1 \nabla(\overline{a}; B) \text{ if } \widetilde{S}, \widetilde{\sigma}, m \circ \pi_S \Vdash^1 \gamma(\overline{a}, B), \quad (3)$$

where π_S is the projection map $\pi_S : \widetilde{S} \to S$. It is then an immediate consequence of (2) that $\nabla(\overline{a}; B)$ is a disjunctive formula.

Given a set A we define $\mathtt{D}_{\mathsf{B}}(A)$ as the set of all formulas $\nabla(\overline{a}; B)$ with $a \in A$ and $B \subseteq A$.

▶ **Theorem 5.** *The collection* $\mathtt{D}_{\mathsf{B}}$ *provides a disjunctive basis for the signature* Σ_{B}.

As far as we know, this result is new. The hardest part in proving it is actually not to show that the language $\mathtt{D}_{\mathsf{B}}$ is distributive over Σ_{B} or that it admits a distributive law (these are easy exercises that we leave to the reader), but to show that formulas in $\mathtt{D}_{\mathsf{B}}(A)$ can be expressed as one-step formulas in $\mathtt{1ML}^+_{\Sigma_{\mathsf{B}}}(A)$. The reason that this is not so easy is subtle; by contrast, it is fairly straightforward to show that formulas in $\mathtt{D}_{\mathsf{B}}(A)$ can be expressed in $\mathtt{1ML}_{\Sigma_{\mathsf{B}}}(A)$, using Ehrenfeucht-Fraïssé games, see e.g. Fontaine & Place [8]. However, a proper disjunctive basis as we have defined it has to consist of *positive* formulas, and this will be crucial for applications to modal fixpoint logics[1].

▶ **Proposition 6.** *Every formula* $\nabla(\overline{a}; B) \in \mathtt{D}_{\mathsf{B}}$ *is one-step equivalent to a formula in* $\mathtt{1ML}_{\Sigma_{\mathsf{B}}}(A)$.

Our main tool in proving this proposition will be Hall's Marriage Theorem, which can be formulated as follows. A *matching* of a bi-partite graph $\mathbb{G} = (V_1, V_2, E)$ is a subset M of E such that no two edges in M share any common vertex. M is said to *cover* V_1 if $\mathsf{Dom}M = V_1$.

▶ **Fact 7** (Hall's Marriage Theorem)**.** *Let* $\mathbb{G}$ *be a finite bi-partite graph,* $\mathbb{G} = (V_1, V_2, E)$. *Then* $\mathbb{G}$ *has a matching that covers* V_1 *iff, for all* $U \subseteq V_1$, $|U| \leq |E[U]|$, *where* $E[U]$ *is the set of vertices in* V_2 *that are adjacent to some element of* U.

[1] The same subtlety appears in Janin & Lenzi [10], where the translation of the language $\mathtt{D}_{\mathsf{B}}$ into $\mathtt{1ML}^+_{\Sigma_{\mathsf{B}}}$ is required to prove that the graded μ-calculus is equivalent, over trees, to monadic second-order logic. Proposition 6 in fact fills a minor gap in this proof.

Proof of Proposition 6. We will show this for the simple case where B is a singleton $\{b\}$. The general case is an immediate consequence of this (consider the substitution $B \mapsto \bigvee B$).

Where $\overline{a} = (a_1, \ldots, a_n)$, define $I := \{1, \ldots, n\}$. For each subset $J \subseteq I$, let χ_J be the formula

$$\chi_J := \Diamond^{|J|} \bigvee_{i \in J} a_i \wedge \Box^{n+1-|J|} (\bigvee_{i \in J} a_i \vee b),$$

and let γ be the conjunction $\gamma := \bigwedge\{\chi_J \mid J \subseteq I\}$. What the formula χ_J says about a Kripkean (finite) one-step model is that at least $|J|$ elements satisfy the disjunction of the set $\{a_i \mid i \in J\}$, while all but at most $n - |J|$ elements satisfy the disjunction of the set $\{a_i \mid i \in J\} \cup \{b\}$. Abbreviating $\nabla(\overline{a};b) := \nabla(\overline{a};\{b\})$, we claim that $\gamma \equiv^1 \nabla(\overline{a};b)$, and to prove this it suffices to consider Kripkean one-step models.

It is straightforward to verify that the formula γ is a semantic one-step consequence of $\nabla(\overline{a};b)$. For the converse, consider a Kripkean one-step model (S, σ, m) in which γ is true. Let K be an index set of size $|S| - n$, and disjoint from I. Clearly then, $|I \cup K| = |I| + |K| = |S|$. Furthermore, let $a_k := b$, for all $k \in K$.

We define a bipartite graph $\mathbb{G} := (V_1, V_2, E)$ by setting $V_1 := I \cup K$, $V_2 := S$, and $E := \{(j, s) \in (I \cup K) \times S \mid a_j \in m(s)\}$. By Hall's Theorem the graph $\mathbb{G}$ has a matching M that covers V_1 (a full proof of this is given in [6]). Since the size of the set V_1 is the same as that of V_2, any matching M of $\mathbb{G}$ that covers V_1 is (the graph of) a bijection between these two sets. Furthermore, it easily follows that such an M restricts to a bijection between I and a subset $\{s_1, ..., s_n\}$ of S such that $a_i \in m(s_i)$ for each $i \in I$, and that $b \in m(t)$ for each $t \notin \{s_1, ..., s_n\}$. Hence $\nabla(\overline{a};b)$ is true in (S, σ, m), as required. ◀

This concludes the proof of Theorem 5.

An example without weak pullback preservation. There are also functors that do not preserve weak pullbacks, but do have a disjunctive basis. As an example of this, consider the subfunctor $\mathsf{P}^{2/3}$ of P^3 given by: $\mathsf{P}_{2/3}S = \{(Z_0, Z_1, Z_2) \mid Z_0 \cap Z_1 \neq \varnothing \text{ or } Z_1 \cap Z_2 \neq \varnothing\}$. While it is easy to show that this functor does not preserve weak pullbacks, the signature Σ_{P^3} (regarded as a set of liftings for $\mathsf{P}_{2/3}$ rather than P^3) still admits a disjunctive basis.

A non-example. Finally, we mention an example of a signature that does not admit any disjunctive basis: the signature Σ consisting of the box- and diamond liftings for M does not have a disjunctive basis. The full proof of this can be found in [6].

4 Disjunctive automata and simulation

We now introduce disjunctive automata, which serve as a coalgebraic generalization of non-deterministic automata for the modal μ-calculus.

▶ **Definition 8.** A Λ-automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$ is said to be *disjunctive* (relative to a disjunctive basis $\mathtt{D}$) if $\Theta(c, a) \in \mathtt{D}(A)$, for all colors $c \in \mathsf{PX}$ and all states $a \in A$.

▶ **Definition 9.** Let $\mathbb{A} = (A, \Theta, \Omega, a_I)$ be a Λ-automaton and let $(\mathbb{S}, s_I)$ be a pointed T-model. A strategy f for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_I)$ is *separating* if for every s in $\mathbb{S}$ there is at most one state a in $\mathbb{A}$ such that the position (a, s) is *f-reachable* (i.e., occurs in some f-guided match). We say that $\mathbb{A}$ *strongly accepts* $(\mathbb{S}, s_I)$, notation: $\mathbb{S}, s_I \Vdash_s \mathbb{A}$ if $\exists$ has a separating winning strategy in the game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_I)$.

Disjunctive automata are very well behaved. For instance, the following result, which can be proved using essentially the same argument as in [21], states a *linear-size* model property.

▶ **Theorem 10.** *Let $\mathbb{A} = (A, \Theta, a_I, \Omega)$ be a disjunctive automaton for a set functor T. If $\mathbb{A}$ accepts some pointed T-model, then it accepts one of which the carrier S satisfies $S \subseteq A$.*

The main property of disjunctive automata, which we will use throughout the remainder of this paper, is the following.

▶ **Proposition 11.** *Let $\mathbb{A}$ be a disjunctive Λ-automaton. Then any pointed T-model which is accepted by $\mathbb{A}$ has a pre-image model which is strongly accepted by $\mathbb{A}$.*

Proof. Let $\mathbb{S} = (S, \sigma, V)$ be a pointed T-model, let $s_I \in S$, and let f be a winning strategy for $\exists$ in the acceptance game $\mathcal{A} := \mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_I)$; without loss of generality we may assume that f is positional. We will construct (i) a pointed T-model (X, ξ, W, x_I), (ii) a tree (X, R) which is rooted at x_I (in the sense that for every $t \in X$ there is a unique R-path from x_I to x) and supports (X, ξ) (in the sense that $\xi(x) \in \mathsf{T}R(x)$, for every $x \in X$), (iii) a morphism $h : (X, \xi, W) \to (S, \sigma, V)$ such that $h(x_I) = s_I$. In addition (X, ξ, W, x_I) will be strongly accepted by $\mathbb{A}$.

In more detail, we will construct all of the above step by step, and by a simultaneous induction we will associate, with each $t \in X$ of depth k, a (partial) f-guided match Σ_t of length $2k + 1$; we will denote the final position of Σ_t as (a_t, s_t), and will define $h(t) := s_t$.

For the base step of the construction we take some fresh object x_I, we define Σ_{x_I} to be the match consisting of the single position (a_I, s_I), and set $h(x_I) := s_I$.

Inductively assume that we are dealing with a node $t \in X$ of depth k, and that Σ_t, a_t and s_t are as described above. Since Σ_t is an f-guided match and f is a winning strategy in $\mathcal{A}$, the pair (a_t, s_t) is a winning position for $\exists$ in $\mathcal{A}$. In particular, the marking $m_t : S \to \mathsf{P}A$ prescribed by f at this position satisfies

$$S, \sigma(s_t), m_t \Vdash^1 \Theta(V^\flat(s_t), a_t).$$

Now by disjunctiveness of the automaton $\mathbb{A}$ there is a set $R(t)$ (that we may take to consist of fresh objects), an object $\xi(t) \in \mathsf{T}R(t)$, an A-marking $m'_t : R(t) \to \mathsf{P}A$ and a map $h_t : R(t) \to S$, such that[2] $|m(u)| = 1$ and $m'_t(u) \subseteq m_t(h_t(u))$ for all $u \in R(t)$, $(\mathsf{T}h_t)\xi(t) = \sigma(s_t)$ and

$$R(t), \xi(t), m'_t \Vdash^1 \Theta(V^\flat(s_t), a_t).$$

Let a_u be the unique object such that $m'_t(u) = \{a_u\}$, define $s_u := h_t(u)$, and put $\Sigma_u := \Sigma_t \cdot m_t \cdot (a_u, s_u)$.

With (X, R, x_I) the tree constructed in this way, and observing that $\xi(t) \in R(t) \subseteq X$, we let ξ be the coalgebra map on X. Taking $h : X \to S$ to be the union $(x_I, s_I) \cup \{h_t \mid t \in X\}$, we can easily verify that h is a surjective coalgebra morphism. Finally, we define the valuation $W : \mathtt{X} \to \mathsf{P}X$ by putting $W(p) := \{x \in X \mid hx \in V(p)\}$.

It remains to show that $\mathbb{A}$ strongly accepts the pointed T-model $(\mathbb{X}, x_I)$, with $\mathbb{X} = (X, \xi, W)$; for this purpose consider the following (positional) strategy f' for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{X})$. At a position $(a, t) \in A \times X$ such that $a \neq a_t$ $\exists$ moves randomly (we may show that such a position will not occur); on the other hand, at a position of the form (a_t, t), the move

[2] To simplify our construction, we strengthen clause (3) in Definition 2. This is not without loss of generality, but we may take care of the general case using a routine extension of the present proof.

suggested by the strategy f' is the marking m'_t. Then it is obvious that f' is a separating strategy; to see that f' is winning from starting position (a_I, x_I), consider an infinite match Σ of $\mathcal{A}(\mathbb{A}, \mathbb{X})@(a_I, x_I)$ (finite matches are left to the reader). It is not hard to see that Σ must be of the form $\Sigma = (a_0, x_0)m'_{x_0}(a_1, x_1)m'_{x_1} \cdots$, where $\Sigma^- = (a_0, h(s_0))m_{x_0}(a_1, h(s_1))m_{x_1} \cdots$ is an f-guided match of $\mathcal{A}$. From this observation it is immediate that Σ is won by $\exists$. ◀

We now come to our main application of disjunctive bases, and fill in the main missing piece in the theory of coalgebraic automata based on predicate liftings: a simulation theorem.

▶ **Theorem 12** (Simulation). *Let Λ be a monotone modal signature for the set functor T and assume that Λ has a disjunctive basis. Then there is an effective construction transforming an arbitrary Λ-automaton $\mathbb{A}$ into an equivalent disjunctive Λ-automaton $\mathsf{sim}(\mathbb{A})$.*

Proof. Assume that $\mathtt{D}$ is a disjunctive basis for Λ, and let $\mathbb{A} = (A, \Theta, \Omega, a_I)$ be a Λ-automaton. Our definition of $\mathsf{sim}(\mathbb{A})$ is rather standard [22], so we will confine ourselves to the definitions. The construction takes place in two steps, a 'pre-simulation' step that produces a disjunctive automaton $\mathsf{pre}(\mathbb{A})$ with a non-parity acceptance condition, and a second 'synchronization' step that turns this nonstandard disjunctive automaton into a standard one.

We define the pre-simulation automaton of $\mathbb{A}$ as the structure $\mathsf{pre}(\mathbb{A}) := (A^\sharp, \Theta^\sharp, NBT_\mathbb{A}, R_I)$, where the carrier of the pre-simulation $\mathsf{pre}(\mathbb{A})$ of $\mathbb{A}$ is the collection $A^\sharp$ of *binary* relations over A, and the initial state R_I is the singleton pair $\{(a_I, a_I)\}$. For its transition function, first define the map $\Theta^\star : A \times \mathsf{PX} \to \mathtt{1ML}^+_\Lambda(A \times A)$ by putting, for $a \in A$ and $c \in \mathsf{PX}$:

$$\Theta^\star(a, c) := \Theta(a, c)[\theta_a],$$

where $\theta_a : A \to \mathtt{Latt}(A \times A)$ is the *tagging* substitution given by $\theta_a : b \mapsto (a, b)$. Now, given a state $R \in A^\sharp$ and color $c \in \mathsf{PX}$, take $\Theta^\sharp(R, c)$ to be an arbitrary but fixed formula in $\mathtt{D}(A^\sharp)$ such that

$$\Theta^\sharp(R, c)[\wedge_{A\times A}] \equiv \bigwedge_{a \in \mathsf{Ran} R} \Theta^\star(a, c).$$

Clearly such a formula exists by our assumption on $\mathtt{D}$ being a disjunctive basis for Λ.

Turning to the *acceptance condition*, define a *trace* on an $A^\sharp$-stream $\rho = (R_n)_{0 \le n < \omega}$ to be an A-stream $\alpha = (a_n)_{0 \le n < \omega}$ with $R_i a_i a_{i+1}$ for all $i \le 0$. Calling such a trace α *bad* if $\max\{\Omega(a) \mid a \text{ occurs infinitely often in } \alpha\}$ is odd, we obtain the acceptance condition of the automaton $\mathsf{pre}(\mathbb{A})$ as the set $NBT_\mathbb{A} \subseteq (A^\sharp)^\omega$ of $A^\sharp$-streams that contain no bad trace.

Finally we produce the simulation of $\mathbb{A}$ by forming a certain kind of product of $\mathsf{pre}(\mathbb{A})$ with $\mathbb{Z}$, where $\mathbb{Z} = (Z, \delta, \Omega', z_I)$ is some deterministic parity stream automaton recognizing the ω-regular language $NBT_\mathbb{A}$. More precisely, we define $\mathsf{sim}(\mathbb{A}) := (A^\sharp \times Z, \Theta'', \Omega'', (R_I, z_I))$ where:

- $\Theta''(R, z) := \Theta^\sharp(R)[(Q, \delta(R, z)/Q \mid Q \in A^\sharp]$ and
- $\Omega''(R, z) := \Omega'(z)$.

The equivalence of $\mathbb{A}$ and $\mathsf{sim}(\mathbb{A})$ can be proved by relatively standard means [22]. ◀

5 Lyndon theorems

Lyndon's classical theorem in model theory provides a syntactic characterization of a semantic property, showing that a formula is *monotone* in a predicate P if and only if it is equivalent to a formula in which P occurs only *positively*. A version of this result for the modal μ-calculus

was proved by d'Agostino and Hollenberg in [3]. Here, we show that their result holds for any μ-calculus based on a signature that admits a disjunctive basis.

We first turn to the one-step version of the Lyndon Theorem, for which we need the following definition; we also recall the substitutions $\wedge_A$ and $\vee_A$ defined in section 2.

▶ **Definition 13.** A *propositional A-type* is a subset of A. For $B \subseteq A$ and $a \in A$, the formulas τ_B and τ_B^{a+} are defined by:

$$\begin{array}{lcl} \tau_B & := & \bigwedge B \wedge \bigwedge\{\neg a \mid a \in A \setminus B\} \\ \tau_B^{a+} & := & \bigwedge B \wedge \bigwedge\{\neg b \mid b \in A \setminus (B \cup \{a\})\} \end{array}$$

We let τ and τ^{a+} denote the maps $B \mapsto \tau_B$ and $B \mapsto \tau_B^{a+}$, respectively.

▶ **Proposition 14.** *Suppose Λ admits a disjunctive basis. Then for any formula α in $\mathtt{1ML}_\Lambda(A)$ there is a one-step equivalent formula of the form $\delta[\vee_{\mathsf{P}A}][\tau]$ for some $\delta \in \mathtt{D}(\mathsf{P}A)$.*

Proof. Let's first check that everything is correctly typed: note that we have $\vee_{\mathsf{P}A} : \mathsf{P}A \to \mathtt{Bool}(\mathsf{P}A)$ and so $\delta[\vee_{\mathsf{P}A}] \in \mathtt{1ML}_\Lambda(\mathsf{P}A)$, and $\tau_{\mathsf{P}A} : \mathsf{P}A \to \mathtt{Bool}(A)$. So $\delta[\vee_{\mathsf{P}A}][\tau] \in \mathtt{1ML}_\Lambda(A)$, as required.

For the normal form proof, first note that we can use boolean duals of the modal operators to push negations down to the zero-step level. Putting the resulting formula in disjunctive normal form, we obtain a disjunction of formulas of the form $\heartsuit_{\lambda_1}\pi_1 \wedge ... \wedge \heartsuit_{\lambda_k}\pi_k$, where $\pi_1, ..., \pi_k \in \mathtt{Bool}(A)$. Repeatedly applying the distributivity of $\mathtt{D}$ over Λ and the distributive law for $\mathtt{D}$, we can rewrite each such disjunct as a formula of the form $\delta[\sigma]$ where $\delta \in \mathtt{D}(\{1, ..., k\})$ and $\sigma : \{1, ..., k\} \to \mathtt{Bool}(A)$ is defined by setting $i \mapsto \pi_i$. Now, just apply propositional logic to rewrite each formula π_i as a disjunction of formulas in $\tau[\mathsf{P}A]$, and we are done. ◀

▶ **Theorem 15** (One-step Lyndon theorem)**.** *Let Λ be a monotone modal signature for the set functor T and assume that Λ has a disjunctive basis. Any $\alpha \in \mathtt{1ML}_\Lambda(A)$, monotone in the variable $a \in A$, is one-step equivalent to some formula in $\mathtt{1ML}_\Lambda(A)$, which is positive in a.*

Proof. By Proposition 14, we can assume that α is of the form $\delta[\vee_{\mathsf{P}A}][\tau]$ for some $\delta \in \mathtt{D}(\mathsf{P}A)$. Clearly it suffices to show that :

$$\delta[\vee_{\mathsf{P}A}][\tau] \equiv^1 \delta[\vee_{\mathsf{P}A}][\tau^{a+}]$$

One direction, from left to right, is easy since $\delta[\vee_{\mathsf{P}A}]$ is a monotone formula in $\mathtt{1ML}_\Lambda(\mathsf{P}A)$, and $[\![\tau_B]\!]^0_m \subseteq [\![\tau_B^{a+}]\!]^0_m$ for each $B \subseteq A$ and each marking $m : X \to \mathsf{P}A$.

For the converse direction, suppose $X, \xi, m \Vdash^1 \delta[\vee_{\mathsf{P}A}][\tau^{a+}]$. We define a $\mathsf{P}A$-marking $m_0 : X \to \mathsf{PP}A$ by setting $m_0(u) := \{B \subseteq A \mid B \preceq_a m(u)\}$, where the relation $\preceq_a$ over $\mathsf{P}A$ is defined by $B \preceq_a B'$ iff $B \setminus \{a\} = B' \setminus \{a\}$, and $a \notin B$ or $a \in B'$. We claim that $X, \xi, m_0 \Vdash^1 \delta[\vee_{\mathsf{P}A}]$. Since $\delta[\vee_{\mathsf{P}A}]$ is a monotone formula, it suffices to check that $[\![\tau_B^{a+}]\!]^0_m \subseteq [\![B]\!]^0_{m_0}$ for each $B \subseteq A$. This follows by just unfolding definitions.

Since δ was disjunctive, so is $\delta[\vee_{\mathsf{P}A}]$, as an easy argument will reveal. So we now find a one-step frame morphism $f : (X', \xi') \to (X, \xi)$, together with a marking $m' : X' \to \mathsf{PP}A$ such that $|m'(u)| \leq 1$ and $m'(u) \subseteq m_0(f(u))$ for all $u \in X'$, and such that $X', \xi', m' \Vdash^1 \delta[\vee_{\mathsf{P}A}]$. We define a new A-marking $m'' : X' \to \mathsf{P}A$ on X' by setting $m''(u) = B$, if $m'(u) = \{B\}$, and $m''(u) = m(f(u))$ if $m'(u) = \emptyset$. Note that, for each $B \subseteq A$, we have $[\![B]\!]^0_{m'} \subseteq [\![\tau_B]\!]^0_{m''}$, so by monotonicity of $\delta[\vee_{\mathsf{P}A}]$ we get $X', \xi', m'' \Vdash^1 \delta[\vee_{\mathsf{P}A}][\tau]$.

If we compare the markings m'' and $m \circ f$, we see that $m''(u) \preceq_a m(f(u))$ for all $u \in X'$. If $m'(u) = \emptyset$, then in fact $m''(u) = m(f(u))$ by definition of m''. If $m'(u) = \{B\}$, then $m''(u) = B \in m'(u) \subseteq m_0(f(u))$, hence $B \preceq_a m(f(u))$ by definition of m_0. Since $\delta[\vee_{\mathsf{P}A}][\tau]$

was monotone with respect to the variable a it follows that $X', \xi', m \circ f \Vdash^1 \delta[\vee_{\mathsf{P}A}][\tau]$ and so $X, \xi, m \Vdash^1 \delta[\vee_{\mathsf{P}A}][\tau]$ by naturality, thus completing the proof of the theorem. ◀

A useful corollary to this theorem is the following.

▶ **Corollary 16.** *Suppose Λ is an expressively complete set of monotone predicate liftings for T. If Λ admits a disjunctive basis, then Λ is Lyndon complete and hence $\mu\mathtt{ML}_\Lambda \equiv \mu\mathtt{ML}_\mathsf{T}$.*

At first glance this proposition (of which a full proof can be found in [6]) may seem trivial, but it is important to see that it is not: given a formula φ of $\mu\mathtt{ML}_\mathsf{T}$, a naive definition of an equivalent formula in $\mu\mathtt{ML}_\Lambda$ would be to apply expressive completeness to simply replace each subformula of the form $\heartsuit_\lambda(\psi_1, ..., \psi_n)$ with an equivalent one-step formula α over $\{\psi_1, ..., \psi_n\}$, using only predicate liftings in Λ. But if this subformula contains bound fixpoint variables, these must still appear positively in α in order for the translation to even produce a grammatically correct formula! We need the stronger condition of *Lyndon completeness* for Λ. We do not know whether expressive completeness entails Lyndon completeness in general, but in the presence of a disjunctive basis, it does: this is a consequence of Theorem 15.

We now turn to our Lyndon Theorems for the full coalgebraic modal (fixpoint) languages. Let $(\mu\mathtt{ML}_\Lambda)^M_p$ and $(\mathtt{ML}_\Lambda)^M_p$ denote the fragments of respectively $\mu\mathtt{ML}$ and $\mathtt{ML}_\Lambda$, consisting of the formulas that are positive in the proposition letter p.

▶ **Theorem 17** (Lyndon Theorem)**.** *There is an effective translation $(\cdot)^M_p : \mu\mathtt{ML}_\Lambda \to (\mu\mathtt{ML}_\Lambda)^M_p$, which restricts to a map $(\cdot)^M_p : \mathtt{ML}_\Lambda \to (\mathtt{ML}_\Lambda)^M_p$, and satisfies that*

$\xi \in \mu\mathtt{ML}$ is monotone in p iff $\xi \equiv \xi^M_p$.

Proof. Due to space limitations, we have to confine ourselves to a sketch. By the equivalence between formulas and Λ-automata and the Simulation Theorem, it suffices to prove the analogous statement for disjunctive coalgebra automata. Given a disjunctive Λ-automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$, we define $\mathbb{A}^M_p$ to be the automaton $(A, \Theta^M_p, \Omega, a_I)$, where

$$\Theta^M_p(c, a) := \begin{cases} \Theta(c, a) & \text{if } p \in c \\ \top & \text{if } p \notin c. \end{cases}$$

Clearly $\mathbb{A}^M_p$ is a disjunctive automaton as well, and it is routine to show that $\mathbb{A}^M_p$ is equivalent to a formula in $\mu\mathtt{ML}_\Lambda$ that is positive in the variable p. Our main claim is then that $\mathbb{A}$ is monotone in p iff $\mathbb{A} \equiv \mathbb{A}^M_p$. Some more details of this proof can be found in the appendix. ◀

▶ Remark. Observe that as a corollary of Theorem 17 and the decidability of the satisfiability problem of $\mu\mathtt{ML}_\Lambda$ [2], it is decidable whether a given formula $\xi \in \mu\mathtt{ML}$ is monotone in p.

6 Uniform Interpolation

Uniform interpolation is a very strong form of the interpolation theorem, first proved for the modal μ-calculus in [3]. It was later generalized to coalgebraic modal logics in [16]. However, the proof crucially relies on non-deterministic automata, and for that reason the generalization in [16] is stated for nabla-based languages. With a simulation theorem for predicate liftings based automata in place, we can prove the uniform interpolation theorem for a large class of μ-calculi based on predicate liftings. Given a set $\mathtt{X}$ of proposition letters and a single proposition letter p, it may be convenient to denote the set $\mathtt{X} \cup \{p\}$ as $\mathtt{X}p$.

▶ **Definition 18.** Given a formula $\varphi \in \mu\mathtt{ML}_\Lambda$, we let $\mathtt{X}_\varphi$ denote the set of proposition letters occurring in φ.

A logic $\mathcal{L}$ with semantic consequence relation $\models$ is said to have the property of *uniform interpolation* if, for any formula $\varphi \in \mathcal{L}$ and any set $\mathtt{X} \subseteq \mathtt{X}_\varphi$ of proposition letters, there is a formula $\varphi_\mathtt{X} \in \mathcal{L}(\mathtt{X})$, effectively constructible from φ, such that

$$\varphi \models \psi \text{ iff } \varphi_\mathtt{X} \models \psi, \tag{4}$$

for every formula $\psi \in \mathcal{L}$ such that $\mathtt{X}_\varphi \cap \mathtt{X}_\psi \subseteq \mathtt{X}$.

To see why this property is called uniform *interpolation*, it is not hard to prove that, if $\varphi \models \psi$, with $\mathtt{X}_\varphi \cap \mathtt{X}_\psi \subseteq \mathtt{X}$, then the formula $\varphi_\mathtt{X}$ is indeed an interpolant in the sense that $\varphi \models \varphi_\mathtt{X} \models \psi$ and $\mathtt{X}_{\varphi_\mathtt{X}} \subseteq \mathtt{X}_\varphi \cap \mathtt{X}_\psi$.

▶ **Theorem 19** (Uniform Interpolation). *Let Λ be a monotone modal signature for the set functor T and assume that Λ has a disjunctive basis. Then both logics $\mathtt{ML}_\Lambda$ and $\mu\mathtt{ML}_\Lambda$ enjoy the property of uniform interpolation.*

Following D'Agostino & Hollenberg [3], we prove Theorem 19 by automata-theoretic means. The key proposition in our proof is Proposition 21 below, which refers to the following construction on disjunctive automata.

▶ **Definition 20.** Let $\mathtt{X}$ be a set of proposition letters not containing the letter p. Given a disjunctive $(\Lambda, \mathtt{X}p)$-automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$, we define the map $\Theta^{\exists p} : A \times \mathsf{P}\mathtt{X} \to \mathtt{D}(A)$ by

$$\Theta^{\exists p}(c, a) := \Theta(c, a) \vee \Theta(c \cup \{p\}, a),$$

and we let $\mathbb{A}^{\exists p}$ denote the $(\Lambda, \mathtt{X})$-automaton $(A, \Theta^{\exists p}, \Omega, a_I)$.

▶ **Proposition 21.** *Let $\mathtt{X} \subseteq \mathtt{Y}$ be sets of proposition letters, both not containing the letter p. Then for any disjunctive $(\Lambda, \mathtt{X}p)$-automaton $\mathbb{A}$ and any pointed T-model $(\mathbb{S}, s_I)$ over $\mathtt{Y}$:*

$$\mathbb{S}, s_I \Vdash \mathbb{A}^{\exists p} \textit{ iff } \mathbb{S}', s'_I \Vdash_s \mathbb{A} \textit{ for some } \mathtt{Y}p\textit{-model } (S', s'_I) \textit{ such that } \mathbb{S}'{\restriction_\mathtt{Y}}, s'_I \rightrightarrows \mathbb{S}, s_I. \tag{5}$$

Proof. We only prove the direction from left to right, leaving the other (easier) direction as an exercise to the reader. For notational convenience we assume that $\mathtt{X} = \mathtt{Y}$.

By Proposition 11 it suffices to assume that $(\mathbb{S}, s_I)$ is *strongly* accepted by $\mathbb{A}^{\exists p}$ and find a subset U of S for which we can prove that $\mathbb{S}[p \mapsto U], s_I \Vdash_s \mathbb{A}$. So let f be a separating winning strategy for $\exists$ in $\mathcal{A}(\mathbb{A}^{\exists p}, \mathbb{S})@(a_I, s_I)$ witnessing that $\mathbb{S}, s_I \Vdash_s \mathbb{A}^{\exists p}$. Call a point $s \in S$ *f-accessible* if there is a state $a \in A$ such that the position (a, s) is f-reachable; since this state is unique by the assumption of strong acceptance we may denote it as a_s. Clearly any position of the form (a_s, s) is winning for $\exists$, and hence by legitimacy of f it holds in particular that

$$S, \sigma(s), m_s \Vdash^1 \Theta^{\exists p}(V^\flat(s), a_s),$$

where $m_s : S \to \mathsf{P}A$ denotes the marking selected by f at position (a_s, s). Recalling that $\Theta^{\exists p}(V^\flat(s), a_s) = \Theta(V^\flat(s), a_s) \vee \Theta(V^\flat(s) \cup \{p\}, a_s)$, we define

$$U := \{s \in S \mid s \text{ is } f\text{-accessible and } S, \sigma(s), m_s \nVdash^1 \Theta(V^\flat(s), a_s)\}.$$

By this we ensure that, for all f-accessible points s:

$$s \notin U \text{ implies } S, \sigma(s), m_s \Vdash^1 \Theta(V^\flat(s), a_s) \tag{6}$$

$$\text{while } s \in U \text{ implies } S, \sigma(s), m_s \Vdash^1 \Theta(V^\flat(s) \cup \{p\}, a_s) \tag{7}$$

Now consider the valuation $V_U := V[p \mapsto U]$, and observe that by this definition we have $V_U^\flat(s) = V^\flat(s)$ if $s \notin U$ while $V_U^\flat(s) = V^\flat(s) \cup \{p\}$ if $s \in U$. Combining this with (6) and (7) we find that

$$S, \sigma(s), m_s \Vdash^1 \Theta(V_U^\flat, a_s)$$

whenever s is f-accessible. In other words, f provides a legitimate move m_s in $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_s, s)$ at any position of the form (a_s, s). From this it is straightforward to derive that f itself is a (separating) winning strategy for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{S}[p \mapsto U])@(a_I, s_I)$, and so we obtain that $\mathbb{S}[p \mapsto U], s_I \Vdash_s \mathbb{A}$ as required. ◀

The remaining part of the argument follows by a fairly standard argument going back to D'Agostino & Hollenberg [3] (see also Marti et alii [16]), with a twist provided by the fact that the 'bisimulation quantifier' here refers to pre-images rather than to bisimilar models.

▶ **Proposition 22.** *Given any proposition letter p, there is a map $\exists p$ on $\mu\mathtt{ML}_\Lambda$, restricting to $\mathtt{ML}_\Lambda$, such that $\mathtt{X}_{\exists p.\varphi} = \mathtt{X}_\varphi \setminus \{p\}$ and, for every pointed $(\mathbb{S}, s_I)$ over a set $\mathtt{Y} \supseteq \mathtt{X}_\varphi$ with $p \notin \mathtt{Y}$:*

$$\mathbb{S}, s_I \Vdash \exists p.\varphi \text{ iff } \mathbb{S}', s_I' \Vdash \varphi \text{ for some } \mathtt{Y}p\text{-model } (S', s_I') \text{ such that } \mathbb{S}'{\restriction_{\mathtt{Y}}}, s_I' \rightrightarrows \mathbb{S}, s_I. \tag{8}$$

Proof. Straightforward by the equivalence between formulas and Λ-automata, the Simulation Theorem, and Proposition 21. ◀

Proof of Theorem 19. With $p_1, \ldots, p_n$ enumerating the proposition letters in $\mathtt{X}_\varphi \setminus \mathtt{X}$, set

$$\varphi_{\mathtt{X}} := \exists p_1 \exists p_2 \cdots \exists p_n.\varphi.$$

Then a relatively routine exercise shows that $\varphi \models \psi$ iff $\varphi_{\mathtt{Y}} \models \psi$, for all formulas $\psi \in \mu\mathtt{ML}_\Lambda$ such that $\mathtt{X}_\varphi \cap \mathtt{X}_\psi \subseteq \mathtt{X}$. Finally, it is not difficult to verify that $\varphi_{\mathtt{Y}}$ is fixpoint-free if φ is so; that is, the uniform interpolants of a formula in $\mathtt{ML}_\Lambda$ also belong to $\mathtt{ML}_\Lambda$. ◀

7 Yoneda representation of disjunctive liftings

It is a well known fact in coalgebraic modal logic that predicate liftings have a neat representation via an application of the Yoneda lemma. This was explored by Schröder in [20], where it was used among other things to prove a characterization theorem for the monotone predicate liftings. Here, we apply the same idea to disjunctive liftings. We shall be working with a slightly generalized notion of predicate lifting here, taking a predicate lifting over a finite set of variables A to be a natural transformation $\lambda : \breve{\mathsf{P}}^A \to \breve{\mathsf{P}} \circ \mathsf{T}$. Clearly, one-step formulas in $\mathtt{1ML}_\Lambda(A)$ can then be viewed as predicate liftings over A.

▶ **Definition 23.** Let $\lambda : \breve{\mathsf{P}}^A \to \breve{\mathsf{P}} \circ \mathsf{T}$ be a predicate lifting over variables $A = \{a_1, ..., a_n\}$. The *Yoneda representation* $y(\lambda)$ of λ is the subset

$$\lambda_{\mathsf{P}A}(\mathsf{true}_{a_1}, ..., \mathsf{true}_{a_n}) \in \mathsf{PTP}A$$

where $\mathsf{true}_{a_i} = \{B \subseteq A \mid a_i \in B\}$. We shall write simply $\lambda \subseteq \mathsf{TP}A$ instead of $y(\lambda)$.

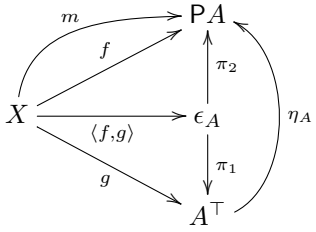
▶ **Definition 24.** Given a set A, let $A^\top$ be the set $A \cup \{\top\}$. Let $\epsilon_A \subseteq A^\top \times \mathsf{P}A$ be the relation defined by $a \epsilon_A B$ iff $a \in B$, and $\top \epsilon_A B$ for all $B \subseteq A$. Let $\eta_A : A^\top \to \mathsf{P}A$ be defined by $\eta_A(a) = \{a\}$, and $\eta_A(\top) = \varnothing$.

In the remainder of this section we assume familiarity with the Barr relation lifting $\overline{\mathsf{T}}$ associated with a functor T; see [14] for the definition and some basic properties.

▶ **Definition 25.** A predicate lifting $\lambda \subseteq \mathsf{T}\mathsf{P}A$ is said to be *divisible* if, for all $\alpha \in \lambda$ there is some $\beta \in \mathsf{T}A^\top$ such that $(\beta, \alpha) \in \overline{\mathsf{T}}(\epsilon_A)$ and $\mathsf{T}\eta_A(\beta) \in \lambda$.

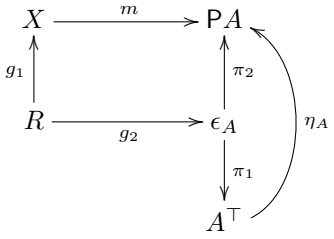
▶ **Proposition 26.** *Any disjunctive lifting over A is divisible, and if T preserves weak pullbacks the disjunctive liftings over A are precisely the divisible ones.*

Proof. Suppose $\lambda \subseteq \mathsf{T}\mathsf{P}A$ is disjunctive, and pick $\alpha \in \lambda$. Then $\mathsf{P}A, \alpha, \mathsf{id}_{\mathsf{P}A} \Vdash^1 \lambda$, so since λ is disjunctive there are some one-step model (X, ξ, m) and map $f : X \to \mathsf{P}A$ with $m : X \to \mathsf{P}A$, $m(u) \subseteq f(u)$ for all $u \in X$, $\mathsf{T}f(\xi) = \alpha$, and $|m(u)| \leq 1$ for all $u \in X$. We define a map $g : X \to A^\top$ by setting $g : u \mapsto \top$ if $m(u) = \varnothing$, $g : u \mapsto a$ if $m(u) = \{a\}$. We tuple the maps f, g to get a map $\langle f, g\rangle : X \to A^\top \times \mathsf{P}A$. In fact, since $m(u) \subseteq f(u)$ for all $u \in X$, we have $\langle f, g\rangle : X \to \epsilon_A$. Let $\pi_1 : \epsilon_A \to A^\top$ and $\pi_2 : \epsilon_A \to \mathsf{P}A$ be the projection maps. We have the following diagram, in which the two triangles and the outer edges commute (i.e., $m = \eta_A \circ g$).



Now apply T to this diagram and define $\beta \in \mathsf{T}A^\top$ to be $\mathsf{T}(\pi_1 \circ \langle f, g\rangle)(\xi) = \mathsf{T}g(\xi)$. First, we have $(\beta, \alpha) \in \overline{\mathsf{T}}(\epsilon_A)$, witnessed by $\mathsf{T}(\langle f, g\rangle)(\xi) \in \mathsf{T}\epsilon_A$. We claim that $\mathsf{T}\eta_A(\beta) \in \lambda$. But since $X, \xi, m \Vdash^1 \lambda$ and $m = \eta_A \circ g$, naturality of λ applied to the map $g : X \to A^\top$, gives $A^\top, \beta, \eta_A \Vdash^1 \lambda$. Another naturality argument, applied to $\eta_A : (A^\top, \beta, \eta_A) \to (\mathsf{P}A, \mathsf{T}\eta_A(\beta), \mathsf{id}_{\mathsf{P}A})$ gives $\mathsf{P}A, \mathsf{T}\eta_A(\beta), \mathsf{id}_{\mathsf{P}A} \Vdash^1 \lambda$, i.e., $\mathsf{T}\eta_A(\beta) \in \lambda$.

For the converse direction, under the assumption that T preserves weak pullbacks, suppose that λ is divisible, and suppose $X, \xi, m \Vdash^1 \lambda$. We get $\mathsf{T}m(\xi) \in \lambda$ and so we find some $\beta \in \mathsf{T}A^\top$ with $\beta(\overline{\mathsf{T}}\epsilon_A)\mathsf{T}m(\xi)$ and $\mathsf{T}\eta_A(\beta) \in \lambda$. Pick some $\beta' \in \mathsf{T}\epsilon_A$ with $\mathsf{T}\pi_2(\beta') = \mathsf{T}m(\xi)$ and $\mathsf{T}\pi_1(\beta') = \beta$. Let R, g_1, g_2 be the pullback of the diagram $X \to \mathsf{P}A \leftarrow \epsilon_A$, shown in the diagram.



By weak pullback preservation there is $\rho \in \mathsf{T}R$ with $\mathsf{T}g_1(\rho) = \xi$ and $\mathsf{T}g_2(\rho) = \beta'$. The map $g_1 : (R, \rho) \to (X, \xi)$ is thus a cover, and we have a marking m' on R defined by $\eta_A \circ \pi_1 \circ g_2$ (follow the bottom-right path in the previous diagram). It is now routine to check that $R, \rho, m' \Vdash^1 \lambda$, and $|m'(u)| \leq 1$ and $m'(u) \subseteq m(g_1(u))$ for all $u \in R$, so we are done. ◀

For the moment, we leave the question open, whether a similar characterization of disjunctive predicate liftings can be proved without weak pullback preservation. We also leave it as an open problem to characterize the functors that admit a disjunctive basis.

References

1 J. Bergfeld. Moss's coalgebraic logic: Examples and completeness results. Master's thesis, Institute for Logic, Language and Computation, University of Amsterdam, 2009.

2 C. Cîrstea, C. Kupke, and D. Pattinson. EXPTIME tableaux for the coalgebraic μ-calculus. In E. Grädel and R. Kahle, editors, *Computer Science Logic (CSL 2009)*, volume 5771 of *Lecture Notes in Computer Science*, pages 179–193. Springer, 2009.

3 G. D'Agostino and M. Hollenberg. Logical questions concerning the μ-calculus. *Journal of Symbolic Logic*, 65:310–332, 2000.

4 S. Enqvist, F. Seifan, and Y. Venema. Completeness for coalgebraic fixpoint logic. In *Proceedings of the 25th EACSL Annual Conference on Computer Science Logic (CSL 2016)*, volume 62 of *LIPIcs*, pages 7:1–7:19, 2016.

5 S. Enqvist, F. Seifan, and Y. Venema. Completeness for μ-calculi: a coalgebraic approach. Technical Report PP-2017-04, Institute for Logic, Language and Computation, Universiteit van Amsterdam, 2017.

6 S. Enqvist and Y. Venema. Disjunctive bases: Normal forms for modal logics. Technical Report PP–2017–05, Institute for Logic, Language and Computation, Universiteit van Amsterdam, 2017. URL: `http://www.illc.uva.nl/Research/Publications/Reports/PP-2017-05.text.pdf`.

7 G. Fontaine, R. Leal, and Y. Venema. Automata for coalgebras: An approach using predicate liftings. In *Automata, Languages and Programming: 37th International Colloquium ICALP'10*, volume 6199 of *LNCS*, pages 381–392. Springer, 2010.

8 G. Fontaine and T. Place. Frame definability for classes of trees in the mu-calculus. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science (MFCS 2010)*, pages 381–392. Springer, 2010.

9 E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logic, and Infinite Games*, volume 2500 of *LNCS*. Springer, 2002.

10 D. Janin and G. Lenzi. Relating levels of the mu-calculus hierarchy and levels of the monadic hierarchy. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science (LICS 2001)*, pages 347–356, 2001.

11 D. Janin and I. Walukiewicz. Automata for the modal μ-calculus and related results. In J. Wiedermann and P. Hájek, editors, *Mathematical Foundations of Computer Science 1995, 20th International Symposium (MFCS'95)*, volume 969 of *LNCS*, pages 552–562. Springer, 1995.

12 D. Janin and I. Walukiewicz. On the expressive completeness of the propositional μ-calculus w.r.t. monadic second-order logic. In *Proceedings of the Seventh International Conference on Concurrency Theory, CONCUR '96*, volume 1119 of *LNCS*, pages 263–277, 1996.

13 D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

14 C. Kupke, A. Kurz, and Y. Venema. Completeness for the coalgebraic cover modality. *Logical Methods in Computer Science*, 8(3), 2010.

15 A. Kurz and R. Leal. Modalities in the stone age: a comparison of coalgebraic logics. *Theoretical Computer Science*, 430:88–116, 2012.

16 J. Marti, F. Seifan, and Y. Venema. Uniform interpolation for coalgebraic fixpoint logic. In Lawrence S. Moss and Pawel Sobocinski, editors, *Proceedings of the Sixth Conference on Algebra and Coalgebra in Computer Science (CALCO 2015)*, volume 35 of *LIPIcs*, pages 238–252. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. URL: `http://www.dagstuhl.de/dagpub/978-3-939897-84-2`.

17 L. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96:277–317, 1999. (Erratum published *APAL* 99:241–259, 1999).

18 E. Pacuit and S. Salame. Majority logic. In *Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR2004)*, pages 598–605, 2004.

19 D. Pattinson. Coalgebraic modal logic: soundness, completeness and decidability of local consequence. *Theoretical Computer Science*, 309(1–3):177–193, 2003.

20 L. Schröder. Expressivity of coalgebraic modal logic: the limits and beyond. *Theoretical Computer Science*, pages 230–247, 2008.

21 Y. Venema. Automata and fixed point logic: a coalgebraic perspective. *Information and Computation*, 204:637–678, 2006.

22 Y. Venema. Lectures on the modal μ-calculus. Lecture Notes, ILLC, University of Amsterdam, 2012.

23 I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. *Information and Computation*, 157:142–182, 2000.

A Universal Construction for (Co)Relations*†

Brendan Fong[1] and Fabio Zanasi[2]

1 **University of Pennsylvania, United States of America**
2 **University College London, United Kingdom**

Abstract

Calculi of string diagrams are increasingly used to present the syntax and algebraic structure of various families of circuits, including signal flow graphs, electrical circuits and quantum processes. In many such approaches, the semantic interpretation for diagrams is given in terms of relations or corelations (generalised equivalence relations) of some kind. In this paper we show how semantic categories of both relations and corelations can be characterised as colimits of simpler categories. This modular perspective is important as it simplifies the task of giving a complete axiomatisation for semantic equivalence of string diagrams. Moreover, our general result unifies various theorems that are independently found in literature and are relevant for program semantics, quantum computation and control theory.

1998 ACM Subject Classification F.3.2 [Semantics of Programming Languages]: Algebraic approaches to semantics

Keywords and phrases corelation, prop, string diagram

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.12

1 Introduction

Network-style diagrammatic languages appear in diverse fields as a tool to reason about computational models of various kinds, including signal processing circuits, quantum processes, Bayesian networks and Petri nets, amongst many others. In the last few years, there have been more and more contributions towards a uniform, formal theory of these languages which borrows from the well-established methods of programming language semantics. A significant insight stemming from many such approaches is that a *compositional* analysis of network diagrams, enabling their reduction to elementary components, is more effective when system behaviour is thought as a *relation* instead of a function.

A paradigmatic case is the one of signal flow graphs, a foundational structure in control theory: a series of recent works [3, 1, 5, 6, 12] gives this graphical language a syntax and a semantics where each signal flow diagram is interpreted as a subspace (a.k.a. linear relation) over streams. The highlight of this approach is a sound and complete axiomatisation for semantic equivalence: what is of interest for us is how this result is achieved in [3], namely through a *modular* account of the domain of subspaces. The construction can be studied for any field k: one considers the prop[1] SV_k whose arrows $n \to m$ are subspaces of $\mathsf{k}^n \times \mathsf{k}^m$, composed as relations. As shown in in [7, 23], SV_k enjoys a universal characterisation: it is

* An extended version of this paper, with an appendix containing omitted proofs, is available at https://arxiv.org/abs/1703.08247/.

† Brendan Fong acknowledges support from the Queen Elizabeth Scholarship, Oxford, and the Basic Research Office of the ASDR&E through ONR N00014-16-1-2010.

1 A prop is a symmetric monoidal category with objects the natural numbers [17]. It is the typical setting for studying both the syntax and the semantics of network diagrams.

the pushout (in the category of props) of props of *spans* and of *cospans* over Vect_k, the prop with arrows $n \to m$ the linear maps $\mathsf{k}^n \to \mathsf{k}^m$:

$$\begin{array}{ccc} \mathsf{Vect}_\mathsf{k} + {\mathsf{Vect}_\mathsf{k}}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{Vect}_\mathsf{k}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{Vect}_\mathsf{k}) & \longrightarrow & \mathsf{SV}_\mathsf{k}. \end{array} \tag{1}$$

In linear algebraic terms, the two factorisation properties expressed by (1) correspond to the representation of a subspace in terms of a basis (span) and the solution set of a system of linear equations (cospan). Most importantly, this picture provides a roadmap towards a complete axiomatisation for SV_k: one starts from the domain Vect_k of linear maps, which is axiomatised by the equations of Hopf algebras, then combines it with its opposite ${\mathsf{Vect}_\mathsf{k}}^{op}$ via two distributive laws of props [16], one yielding an axiomatisation for $\mathsf{Span}(\mathsf{Vect}_\mathsf{k})$ and the other one for $\mathsf{Cospan}(\mathsf{Vect}_\mathsf{k})$. Finally, merging these two axiomatisations yields a complete axiomatisation for SV_k, called the theory of interacting Hopf algebras [7, 23].

It was soon realised that this modular construction was of independent interest, and perhaps evidence of a more general phenomenon. In [24] it is shown that a similar construction could be used to characterise the prop ER of equivalence relations, using as ingredients In, the prop of injections, and F, the prop of total functions. The same result is possible by replacing equivalence relations with partial equivalence relations and functions with partial functions, forming a prop PF. In both cases, the universal construction yields a privileged route to a complete axiomatisation, of ER and of PER respectively [24].

$$\begin{array}{ccc} \mathsf{In} + \mathsf{In}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{In}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{F}) & \longrightarrow & \mathsf{ER} \end{array} \qquad \begin{array}{ccc} \mathsf{In} + \mathsf{In}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{In}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{PF}) & \longrightarrow & \mathsf{PER}. \end{array} \tag{2}$$

Even though a pattern emerges, it is certainly non-trivial: for instance, if one naively mimics the linear case (1) in the attempt of characterising the prop of relations, the construction collapses to the terminal prop $\mathbf{1}$.

$$\begin{array}{ccc} \mathsf{F} + \mathsf{F}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{F}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{F}) & \longrightarrow & \mathbf{1} \end{array} \tag{3}$$

More or less at the same time, diagrammatic languages for various families of circuits, including linear time-invariant dynamical systems [12], were analysed using so-called *corelations*, which are generalised equivalence relations [10, 9, 11, 2]. Even though they were not originally thought of as arising from a universal construction like the examples above, corelations still follow a modular recipe, as they are expressible as a quotient of $\mathsf{Cospan}(\mathcal{C})$, for some prop $\mathcal{C}$. Thus by analogy we can think of them as yielding one half of the diagram

$$\begin{array}{ccc} \mathcal{C} + \mathcal{C}^{op} & & \\ \downarrow & & \\ \mathsf{Cospan}(\mathcal{C}) & \longrightarrow & \mathsf{Corel}(\mathcal{C}). \end{array} \tag{4}$$

In this paper we clarify the situation by giving a unifying perspective for all these constructions.

We prove a general result, which

- implies (1) and (2) as special cases;
- explains the failure of (3);
- extends (4) to a pushout recipe for corelations.

More precisely, our theorem individuates sufficient conditions for characterising the category $\mathsf{Rel}(\mathcal{C})$ of $\mathcal{C}$-relations as a pushout. A dual construction yields the category $\mathsf{Corel}(\mathcal{C})$ of $\mathcal{C}$-corelations as a pushout. For the case of interest when $\mathcal{C}$ is a prop, the two constructions look as follows.

$$\begin{array}{ccc} \mathcal{A}+\mathcal{A}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{C}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathcal{A}) & \longrightarrow & \mathsf{Rel}(\mathcal{C}). \end{array} \qquad \begin{array}{ccc} \mathcal{A}+\mathcal{A}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{A}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathcal{C}) & \longrightarrow & \mathsf{Corel}(\mathcal{C}). \end{array} \tag{5}$$

The variant ingredient $\mathcal{A}$ is a subcategory of $\mathcal{C}$. In order to make the constructions possible, $\mathcal{A}$ has to satisfy certain requirements in relation with the factorisation system $(\mathcal{E}, \mathcal{M})$ on $\mathcal{C}$ which defines $\mathcal{C}$-relations (as jointly-in-$\mathcal{M}$ spans) and $\mathcal{C}$-corelations (as jointly-in-$\mathcal{E}$ cospans). For instance, taking $\mathcal{A}$ to be $\mathcal{C}$ itself succeeds in (1) (and in fact, for any abelian $\mathcal{C}$), but fails in (3).

Besides explaining existing constructions, our result opens the lead for new applications. In particular, we observe that under mild conditions the construction lifts to the category $\mathcal{C}^T$ of T-algebras for a monad $T\colon \mathcal{C} \to \mathcal{C}$. We leave the exploration of this and other ramifications for future work.

Synopsis

Section 2 introduces the necessary preliminaries about factorisation systems and (co)relations, and shows the subtleties of mapping spans into corelations in a functorial way. Section 3 states our main result and some of its consequences. We first formulate the construction for categories (Theorem 11), and then for props (Theorem 18), which are our prime object of interest in applications. Section 4 is devoted to show various instances of our construction. We illustrate the case of equivalence relations, of partial equivalence relations, of subspaces, of linear corelations, and finally of relations of algebras. Finally, Section 5 summarises our contribution and looks forward to further work. The extended version of this paper contains an appendix detailing the proofs of Theorems 11 and 18.

Conventions

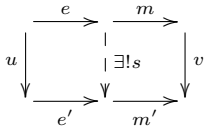
We write $f\,;g$ for composition of $f\colon X \to Y$ and $g\colon Y \to Z$ in a category $\mathcal{C}$. It will be sometimes convenient to indicate an arrow $f\colon X \to Y$ of $\mathcal{C}$ as $X \xrightarrow{f\in\mathcal{C}} Y$ or also $\xrightarrow{\in\mathcal{C}}$, if names are immaterial. Also, we write $X \xleftarrow{f\in\mathcal{C}} Y$ for an arrow $X \xrightarrow{f\in\mathcal{C}^{op}} Y$. We use $\oplus$ for the monoidal product in a monoidal category, with unit object I. Monoidal categories and functors will be strict when not stated otherwise.

2 (Co)relations

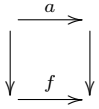
In this section we review the categorical approach to relations, based on the observation that in Set they are the jointly mono spans. We introduce in parallel the dual notion, called corelations [10]: these are jointly epi cospans and can be seen as an abstraction of the concept of equivalence relation.

▶ **Definition 1.** A **factorisation system** $(\mathcal{E}, \mathcal{M})$ in a category $\mathcal{C}$ comprises subcategories $\mathcal{E}$, $\mathcal{M}$ of $\mathcal{C}$ such that

1. $\mathcal{E}$ and $\mathcal{M}$ contain all isomorphisms of $\mathcal{C}$.
2. every morphism $f \in \mathcal{C}$ admits a factorisation $f = e; m$, $e \in \mathcal{E}$, $m \in \mathcal{M}$.
3. given f, f', with factorisations $f = e; m$, $f' = e'; m'$ of the above sort, for every u, v such that $f; v = u; f'$ there exists a unique s making the following diagram commute.



▶ **Definition 2.** Given a category $\mathcal{C}$, we say that a subcategory $\mathcal{A}$ is **stable under pushout** if for every pushout square



such that $a \in \mathcal{A}$, we also have that $f \in \mathcal{A}$. Similarly, we say that $\mathcal{A}$ is **stable under pullback** if for every pullback square labelled as above $f \in \mathcal{A}$ implies $a \in \mathcal{A}$.

A factorisation system $(\mathcal{E}, \mathcal{M})$ is **stable** if $\mathcal{E}$ is stable under pullback, **costable** if $\mathcal{M}$ is stable under pushout, and **bistable** if it is both stable and costable.

Examples of bistable factorisation systems include the trivial factorisation systems $(\mathcal{I}_\mathcal{C}, \mathcal{C})$ and $(\mathcal{C}, \mathcal{I}_\mathcal{C})$ in any category $\mathcal{C}$, where $\mathcal{I}_\mathcal{C}$ is the subcategory containing exactly the isomorphisms in $\mathcal{C}$, the epi-mono factorisation system in any topos, or the epi-mono factorisation system in any abelian category. Stable factorisation systems include the (regular epi, mono) factorisation system in any regular category, such as any category monadic over Set. Dually, costable factorisation systems include the (epi, regular mono) factorisation system in any coregular category, such as the category of topological spaces and continuous maps.

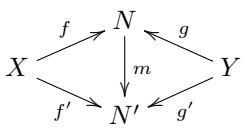
▶ **Definition 3.**

- Given a category $\mathcal{C}$ with pushouts, the category $\mathsf{Cospan}(\mathcal{C})$ has the same objects as $\mathcal{C}$ and arrows $X \to Y$ isomorphism classes of cospans $X \xrightarrow{f} \xleftarrow{g} Y$ in $\mathcal{C}$. The composite of $X \xrightarrow{f} \xleftarrow{g} Y$ and $Y \xrightarrow{h} \xleftarrow{i} Z$ is obtained by taking the pushout of $\xleftarrow{g} \xrightarrow{h}$.
- Given a category $\mathcal{C}$ with pullbacks, the category $\mathsf{Span}(\mathcal{C})$ has the same objects as $\mathcal{C}$ and arrows $X \to Y$ isomorphism classes of spans $X \xleftarrow{f} \xrightarrow{g} Y$ in $\mathcal{C}$. The composite of $X \xleftarrow{f} \xrightarrow{g} Y$ and $Y \xleftarrow{h} \xrightarrow{i} Z$ is obtained by taking the pullback of $\xrightarrow{g} \xleftarrow{h}$.

When $\mathcal{C}$ also has a (co)stable factorisation system, we may define a category of (co)relations with respect to this system.

▶ **Definition 4.**

- Given a category $\mathcal{C}$ with pushouts and a costable factorisation system $(\mathcal{E}, \mathcal{M})$, the category $\mathsf{Corel}(\mathcal{C})$ has the same objects as $\mathcal{C}$. The arrows $X \to Y$ are equivalence classes of cospans $X \xrightarrow{f} N \xleftarrow{g} Y$ under the symmetric, transitive closure of the following relation: two cospans $X \xrightarrow{f} N \xleftarrow{g} Y$ and $X \xrightarrow{f'} N' \xleftarrow{g'} Y$ are related if there exists $N \xrightarrow{m} N'$ in $\mathcal{M}$ such that



(6)

commutes. This notion of equivalence respects composition of cospans, and so $\mathsf{Corel}(\mathcal{C})$ is indeed a category. We call the morphisms in this category **corelations**.

- Given a category $\mathcal{C}$ with pullbacks and a stable factorisation system, we can dualise the above to define the category $\mathsf{Rel}(\mathcal{C})$ of **relations**.

(Co)stability is needed in order to ensure that composition of (co)relations is associative, *cf.* [10, §3.3]. For proofs it is convenient to give an alternative description of (co)relations.

▶ **Proposition 5.** When $\mathcal{C}$ has binary coproducts, corelations are in one-to-one correspondence with isomorphism classes of cospans such that the copairing $[p, q] \colon X + Y \to N$ lies in $\mathcal{E}$.

When $\mathcal{C}$ has binary products, relations are in one-to-one correspondence with isomorphism classes of spans such that the pairing $\langle f, g \rangle \colon N \to X \times Y$ lies in $\mathcal{M}$.

We refer to Appendix A.1 of the extended version for a proof of the proposition.

We call a span $\overset{f}{\leftarrow} \overset{g}{\rightarrow}$ **jointly-in-$\mathcal{M}$** if the pairing $\langle f, g \rangle$ lies in $\mathcal{M}$, and analogously for $\mathcal{E}$ and for cospans. To each relation there is thus, up to isomorphism, a canonical representation as a jointly-in-$\mathcal{M}$ span, and similarly to each corelation a jointly-in-$\mathcal{E}$ cospan.

▶ **Example 6.** Many examples of relations and corelations are already familiar.

- The category Set is bicomplete and has a bistable epi-mono factorisation system. Relations with respect to this factorisation system are simply the usual binary relations, while corelations from $X \to Y$ in Set are surjective functions $X+Y \to N$; thus their isomorphism classes—the arrows of $\mathsf{Corel}(\mathsf{Set})$—are partitions, or equivalence relations on $X + Y$.
- The category of vector spaces over a field k is abelian, and hence bicomplete with a bistable epi-mono factorisation system. The categories of relations and corelations are isomorphic: a morphism $X \to Y$ in these categories can be thought of as a linear relations, i.e. a subspace of $X \times Y$.
- In any category $\mathcal{C}$ the trivial morphism-isomorphism factorisation system $(\mathcal{C}, \mathcal{I}_{\mathcal{C}})$ is bistable. Relations with respect to $(\mathcal{C}, \mathcal{I}_{\mathcal{C}})$ are equivalence classes of isomorphisms $N \xrightarrow{\sim} X \times Y$, and hence there is a unique relation between any two objects. Corelations are just cospans.
- Dually, relations with respect to the isomorphism-morphism factorisation $(\mathcal{I}_{\mathcal{C}}, \mathcal{C})$ are just spans, and there is a unique corelation between any two objects.

We now study the functorial interpretation of cospans and spans as corelations. This discussion is instrumental in our universal construction for corelations (Theorem 11).

First, given two categories with the same collections of objects, we may speak of **identity-on-objects (ioo)** functors between them, i.e. functors that are the identity map on objects. Four examples of such functors will become relevant in the next section:

$$\begin{array}{lll} \mathcal{C} \to \mathsf{Cospan}(\mathcal{C}) \text{ maps } \overset{f}{\rightarrow} \text{ to } \overset{f}{\rightarrow}\overset{id}{\leftarrow} & & \mathcal{C} \to \mathsf{Span}(\mathcal{C}) \text{ maps } \overset{f}{\rightarrow} \text{ to } \overset{id}{\leftarrow}\overset{f}{\rightarrow} \\ \mathcal{C}^{op} \to \mathsf{Cospan}(\mathcal{C}) \text{ maps } \overset{g}{\leftarrow} \text{ to } \overset{id}{\rightarrow}\overset{g}{\leftarrow} & & \mathcal{C}^{op} \to \mathsf{Span}(\mathcal{C}) \text{ maps } \overset{g}{\leftarrow} \text{ to } \overset{g}{\leftarrow}\overset{id}{\rightarrow} \end{array} \tag{7}$$

We are now ready to discuss the canonical map from cospans to corelations. This is simple: one just interprets a cospan representative as its corelation equivalence class.

▶ **Definition 7.** Let $\mathcal{C}$ be a category equipped with a costable factorisation system $(\mathcal{E}, \mathcal{M})$. We define $\Gamma \colon \mathsf{Cospan}(\mathcal{C}) \to \mathsf{Corel}(\mathcal{C})$ as the ioo functor mapping the isomorphism class of cospans represented by $X \overset{f}{\rightarrow} N \overset{g}{\leftarrow} Y$ to the corelation represented by this cospan.

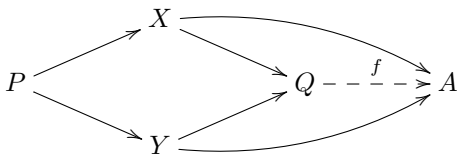
It is straightforward to check that this is well-defined. Moreover,

▶ **Proposition 8.** $\Gamma \colon \mathsf{Cospan}(\mathcal{C}) \to \mathsf{Corel}(\mathcal{C})$ is full.

Proof. Let a be a corelation. Then choosing some representative $X \to N \leftarrow Y$ of a gives a cospan whose Γ-image is a. ◀

Mapping spans to corelations is subtler. Given a span, we may obtain a cospan by taking its pushout. When $\mathcal{C}$ has pushouts and pullbacks, this defines a function on morphisms $\mathsf{Span}(\mathcal{C}) \to \mathsf{Cospan}(\mathcal{C})$. This function is rarely, however, a functor: it may fail to preserve composition. To turn it into a functor, two tweaks are needed: first, we restrict to a subcategory $\mathsf{Span}(\mathcal{A})$ of $\mathsf{Span}(\mathcal{C})$, for some carefully chosen subcategory $\mathcal{A} \subseteq \mathcal{C}$, and second, we take the jointly-in-$\mathcal{E}$ part of the pushout. We call the resulting functor Π, as it takes the *pushout* and then *projects*.

How do we choose $\mathcal{A}$? Given a cospan $X \to A \leftarrow Y$, we may take its pullback to obtain a span $X \leftarrow P \to Y$, and then pushout this span in $\mathcal{C}$ to obtain a cospan $X \to Q \leftarrow Y$. This gives a diagram

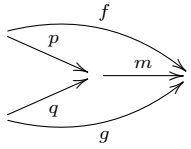


where the map f exists and is unique by the universal property of the pushout. We want this map f to lie in $\mathcal{M}$: by Definition 4, this implies that $X \to A \leftarrow Y$ and $X \to Q \leftarrow Y$ represent the same corelation. This condition is reminiscent of that introduced by Meisen in her work on so-called categories of pullback spans [19].

Note that this pullback and pushout take place in $\mathcal{C}$. We nonetheless ask $\mathcal{A}$ to be closed under pullback, so spans $\xleftarrow{f\in\mathcal{A}}\xrightarrow{g\in\mathcal{A}}$ do indeed form a subcategory $\mathsf{Span}(\mathcal{A})$ of $\mathsf{Span}(\mathcal{C})$.[2]

▶ **Proposition 9.** Let $\mathcal{C}$ be a category equipped with a costable factorisation system $(\mathcal{E}, \mathcal{M})$. Let $\mathcal{A}$ be a subcategory of $\mathcal{C}$ containing all isomorphisms and stable under pullback. Further suppose that the canonical map given by the pushout of the pullback of a cospan in $\mathcal{A}$ lies in $\mathcal{M}$. Then mapping a span in $\mathcal{A}$ to the jointly-in-$\mathcal{E}$ part of its pushout cospan defines an ioo functor $\Pi\colon \mathsf{Span}(\mathcal{A}) \to \mathsf{Corel}(\mathcal{C})$.

Proof. Recall that $\mathsf{Span}(\mathcal{A})$ is generated by morphisms of the form $\xleftarrow{id}\xrightarrow{f\in\mathcal{A}}$ and $\xleftarrow{f\in\mathcal{A}}\xrightarrow{id}$. It is thus enough to show Π preserves composition on arrows of these two types. There exist four cases: (i) $\xleftarrow{id}\xrightarrow{f}\xleftarrow{id}\xrightarrow{g}$, (ii) $\xleftarrow{f}\xrightarrow{id}\xleftarrow{g}\xrightarrow{id}$, (iii) $\xleftarrow{f}\xrightarrow{id}\xleftarrow{id}\xrightarrow{g}$, and (iv) $\xleftarrow{id}\xrightarrow{f}\xleftarrow{g}\xrightarrow{id}$. The first three cases are straightforward to prove, and in fact hold when mapping $\mathsf{Span}(\mathcal{C}) \to \mathsf{Cospan}(\mathcal{C})$. It is the case (iv) that needs our restriction to $\mathsf{Span}(\mathcal{A})$. There $\Pi(\xleftarrow{id}\xrightarrow{f})\Pi(\xleftarrow{g}\xrightarrow{id})$ is represented by the cospan $\xrightarrow{f}\xleftarrow{g}$, while $\Pi(\xleftarrow{id}\xrightarrow{f}\xleftarrow{g}\xrightarrow{id})$ is the represented by the pushout $\xrightarrow{p}\xleftarrow{q}$ of the pullback of $\xrightarrow{f}\xleftarrow{g}$. But by hypothesis, there exists a unique $\xrightarrow{m\in\mathcal{M}}$ making the following diagram commute.



[2] Calling this subcategory $\mathsf{Span}(\mathcal{A})$ is a slight abuse of notation: it may be the case that $\mathcal{A}$ itself has pullbacks, and we have not proved that these agree with pullbacks in $\mathcal{C}$. Nonetheless, this conflict does not cause trouble in any of our examples below, and we stick to this convention for notational simplicity.

This implies that $\xrightarrow{p}\xleftarrow{q}$ and $\xrightarrow{f}\xleftarrow{g}$ represent the same corelation, and so Π is functorial. ◀

For example, if the category $\mathcal{M}$ has pullbacks and these coincide with pullbacks in $\mathcal{C}$, then we can take $\mathcal{A} = \mathcal{M}$. If $\mathcal{C}$ is abelian, we can take $\mathcal{A} = \mathcal{C}$.

3 Main theorem: a universal property for (co)relations

This section states our main result and some consequences. We first fix our ingredients.

▶ **Assumption 10.** Let $\mathcal{C}$ be a category with
- pushouts and pullbacks;
- a costable factorisation system $(\mathcal{E}, \mathcal{M})$ with $\mathcal{M}$ a subcategory of the monos in $\mathcal{C}$;
- a subcategory $\mathcal{A}$ of $\mathcal{C}$ containing $\mathcal{M}$, stable under pullback, and such that the canonical map given by the pushout of the pullback of a cospan in $\mathcal{A}$ lies in $\mathcal{M}$.

Building on the results of Section 2, the second requirement above allows us to form a category $\mathsf{Corel}(\mathcal{C})$ of corelations, whereas the third yields a functor $\Pi\colon \mathsf{Span}(\mathcal{A}) \to \mathsf{Corel}(\mathcal{C})$. We shall also use the functor $\Gamma\colon \mathsf{Cospan}(\mathcal{C}) \to \mathsf{Corel}(\mathcal{C})$ (Definition 7) and a category $\mathcal{A}+_{|\mathcal{A}|}\mathcal{A}^{op}$: its objects are those of $\mathcal{A}$ and the morphisms $X \to Y$ are 'zigzags' $X \xrightarrow{f}\xleftarrow{g}\xrightarrow{h} \dots \xleftarrow{k} Y$ in $\mathcal{A}$. There are ioo functors from $\mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op}$ to $\mathsf{Cospan}(\mathcal{C})$ and to $\mathsf{Span}(\mathcal{C})$, defined on morphisms by taking colimits, respectively limits of zigzags—equivalently, they are defined by pointwise application of the functors in (7). [3] We make all these components interact in our main theorem.

▶ **Theorem 11.** *Let $\mathcal{C}$ and $\mathcal{A}$ be as in Assumption 10. Then the following is a pushout in* **Cat***:*

$$\begin{array}{ccc} \mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{A}) \\ \downarrow & & \downarrow \Pi \\ \mathsf{Cospan}(\mathcal{C}) & \xrightarrow[\Gamma]{} & \mathsf{Corel}(\mathcal{C}) \end{array} \qquad (\star)$$

We leave a complete proof of this theorem to Appendix A.2 of the extended version of this paper. In a nutshell, the key point is that, in light of (6), $\mathsf{Corel}(\mathcal{C})$ differs from $\mathsf{Cospan}(\mathcal{C})$ precisely because it has the extra equations $\xrightarrow{m}\xleftarrow{m} = \xrightarrow{id}\xleftarrow{id}$, with $\xrightarrow{m} \in \mathcal{M}$. But these equations arise by pullback squares in $\mathcal{A}$, and so are equations of zigzags in $\mathsf{Span}(\mathcal{A})$ (*cf.* (8) below). Moreover, the remaining equations of $\mathsf{Span}(\mathcal{A})$ can be generated by using these together with a subset of the equations of $\mathsf{Cospan}(\mathcal{C})$. Hence adding the equations of $\mathsf{Span}(\mathcal{A})$ to those of $\mathsf{Cospan}(\mathcal{C})$ gives precisely $\mathsf{Corel}(\mathcal{C})$, and we have a pushout square.

We now discuss some observations, consequences and examples.

▶ **Remark 12.** If any such $\mathcal{A}$ exists, then we may always take $\mathcal{A} = \mathcal{M}$ and the theorem holds. We record the above, more general, theorem as it explains preliminary results in this direction already in the literature; see the abelian case and examples for details.

Next, we formulate the dual version of the theorem, which yields a characterisation for relations. It is based on a dual version of Assumption 10.

[3] More abstractly, one can see $\mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op}$ as the pushout of $\mathcal{A}$ and $\mathcal{A}^{op}$ over the respective inclusions of $|\mathcal{A}|$, the discrete category on the objects of $\mathcal{A}$. The functors $\mathsf{Span}(\mathcal{A}) \leftarrow \mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op} \to \mathsf{Cospan}(\mathcal{C})$ are then those given by the universal property with respect to (suitable restrictions of) the functors in (7).

▶ **Assumption 13.** Let $\mathcal{C}$ be a category with
- pushouts and pullbacks;
- a stable factorisation system $(\mathcal{E}, \mathcal{M})$ with $\mathcal{E}$ a subcategory of the epis in $\mathcal{C}$;
- a subcategory $\mathcal{A}$ of $\mathcal{C}$ containing $\mathcal{E}$, stable under pushout, and such that the canonical map given by the pullback of the pushout of a span in $\mathcal{A}$ lies in $\mathcal{E}$.

▶ **Corollary 14** (Dual case)**.** Let $\mathcal{C}$ and $\mathcal{A}$ be as in Assumption 13. Then the following is a pushout square in **Cat**.

$$\begin{array}{ccc} \mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op} & \longrightarrow & \mathsf{Cospan}(\mathcal{A}) \\ \downarrow & & \downarrow \\ \mathsf{Span}(\mathcal{C}) & \longrightarrow & \mathsf{Rel}(\mathcal{C}) \end{array} \qquad (\circ)$$

Proof. This corollary is obtained by noting that, given a stable factorisation system $(\mathcal{E}, \mathcal{M})$ in $\mathcal{C}$, with $\mathcal{E}$ a subcategory of the epis, we have a costable factorisation system $(\mathcal{M}^{op}, \mathcal{E}^{op})$ in $\mathcal{C}^{op}$, with $\mathcal{E}^{op}$ a subcategory of the monos. Proposition 9 then gives a functor $\mathsf{Cospan}(\mathcal{A}) = \mathsf{Span}(\mathcal{A}^{op}) \longrightarrow \mathsf{Rel}(\mathcal{C}) = \mathsf{Corel}(\mathcal{C}^{op})$. Noting also that $\mathcal{A} +_{|\mathcal{A}|} \mathcal{A}^{op} = \mathcal{A}^{op} +_{|\mathcal{A}|} (\mathcal{A}^{op})^{op}$ and $\mathsf{Span}(\mathcal{C}) = \mathsf{Cospan}(\mathcal{C}^{op})$, we can hence apply Theorem 11. ◀

As a notable instance of Theorem 11, we can specialise to the case of abelian categories and their epi-mono factorisation system. In this case we can simply pick $\mathcal{A}$ to be $\mathcal{C}$ itself.

▶ **Corollary 15** (Abelian case)**.** Let $\mathcal{C}$ be an abelian category. Then the following is a pushout square in **Cat**:

$$\begin{array}{ccc} \mathcal{C} +_{|\mathcal{C}|} \mathcal{C}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{C}) \\ \downarrow & & \downarrow \Pi \\ \mathsf{Cospan}(\mathcal{C}) & \xrightarrow[\Gamma]{} & \mathsf{Corel}(\mathcal{C}) \cong \mathsf{Rel}(\mathcal{C}) \end{array} \qquad (\triangle)$$

where we take (co)relations with respect the epi-mono factorisation system.

Proof. As $\mathcal{C}$ is abelian, it is finitely bicomplete and has a bistable factorisation system given by epis and monos. Furthermore, we need not restrict our spans to some subcategory $\mathcal{A}$: in an abelian category the pullback of a cospan $X \xrightarrow{f} A \xleftarrow{g} Y$ can be computed via the kernel of the joint map $X \oplus Y \xrightarrow{[f,-g]} A$, and similarly pushouts can be computed via cokernel, whence the canonical map from the pushout of the pullback of a given cospan to itself is always mono, being the inclusion of the image of the joint map into the apex. Similarly, the map from a span to the pullback of its pushout is simply the joint map with codomain restricted to its image, and hence always epi. Thus $\mathcal{C}$ meets both Assumptions 10 and 13 with $\mathcal{A} = \mathcal{C}$. Then the category of corelations is the pushout of the span $\mathsf{Cospan}(\mathcal{C}) \leftarrow \mathcal{C} +_{|\mathcal{C}|} \mathcal{C}^{op} \rightarrow \mathsf{Span}(\mathcal{C})$. But by the dual theorem (Corollary 14), the pushout of this span is also the category of relations. Thus the two categories are isomorphic. Explicitly, the isomorphism is given by taking a corelation to the jointly mono part of its pullback span, and taking a relation to the jointly epi part of its pushout cospan. ◀

▶ **Remark 16.** In Theorem 11, the diagram $\mathsf{Cospan}(\mathcal{C}) \leftarrow \mathcal{A} +_{|\mathcal{A}|} \mathcal{A} \rightarrow \mathsf{Span}(\mathcal{A})$ 'knows' only about $\mathcal{M}$, not the factorisation system $(\mathcal{E}, \mathcal{M})$. This is enough, however, since if $\mathcal{M}$ is part of a factorisation system, then the factorisation system is unique.

Indeed, suppose we have $\mathcal{E}$, $\mathcal{E}'$ such that both $(\mathcal{E}, \mathcal{M})$ and $(\mathcal{E}', \mathcal{M})$ are factorisation systems. Take $e \in \mathcal{E}$. Then the factorisation system $(\mathcal{E}', \mathcal{M})$ gives a factorisation $e = e'; m_1$, while $(\mathcal{E}, \mathcal{M})$ gives a factorisation $e' = e_2; m_2$. By substitution, we have $e = e_2; m_2; m_1$. By uniqueness of factorisation, we can then assume without loss of generality that $e = e_2$ and $m_2; m_1 = id$. Next, using $e = e_2$ and substitution in $e' = e_2; m_2$, we similarly arrive at $m_1; m_2 = id$. Thus m_1 is an isomorphism, and hence lies in $\mathcal{E}'$. This implies that $e = e'; m_1 \in \mathcal{E}'$, and hence $\mathcal{E} \subseteq \mathcal{E}'$. We may similarly show that $\mathcal{E}' \subseteq \mathcal{E}$, and hence that the two categories are equal.

The next corollary is instrumental in giving categories of (co)relations a presentation by generators and equations.

▶ **Corollary 17.** Suppose $\mathcal{A}$ and $\mathcal{C}$ are as in Assumption 10. Then $\mathsf{Corel}(\mathcal{C})$ is freely generated by the objects of $\mathcal{C}$ and arrows $\xrightarrow{f}$, $\xleftarrow{g}$ of $\mathcal{C}$ quotiented by

$$\xrightarrow{f \in \mathcal{A}} \xleftarrow{g \in \mathcal{A}} = \xleftarrow{p \in \mathcal{A}} \xrightarrow{q \in \mathcal{A}} \qquad \text{whenever } \xleftarrow{p}\xrightarrow{q} \text{ pulls back } \xrightarrow{f}\xleftarrow{g} \tag{8}$$

$$\xleftarrow{f \in \mathcal{C}} \xrightarrow{g \in \mathcal{C}} = \xrightarrow{p \in \mathcal{C}} \xleftarrow{q \in \mathcal{C}} \qquad \text{whenever } \xrightarrow{p}\xleftarrow{q} \text{ pushes out } \xleftarrow{f}\xrightarrow{g}. \tag{9}$$

Equivalently, $\mathsf{Corel}(\mathcal{C})$ is the quotient of $\mathsf{Cospan}(\mathcal{C})$ by (8). A dual statement holds for $\mathsf{Rel}(\mathcal{C})$.

Note that, in light of Remark 12, one may also replace (8) by the subset of axioms

$$\xrightarrow{f \in \mathcal{M}} \xleftarrow{g \in \mathcal{M}} = \xleftarrow{p \in \mathcal{M}} \xrightarrow{q \in \mathcal{M}} \qquad \text{whenever } \xleftarrow{p}\xrightarrow{q} \text{ pulls back } \xrightarrow{f}\xleftarrow{g}.$$

As $\mathcal{M} \subseteq \mathcal{A}$ by Assumption 10, this may give a smaller presentation.

The importance of the above observation stems from the fact that sets of equations (8) and (9) yield a presentation for categories of spans and cospans over $\mathcal{C}$ respectively. In various interesting cases, they enjoy a *finitary* axiomatisation, which can be elegantly described in terms of distributive laws between categories [20, 16]. Under this light, Corollary 17 provides a recipe for axiomatising categories of (co)relations starting from existing results about spans and cospans. For instance, this is the strategy adopted in the literature to axiomatise finite equivalence relations [24, 9], finite partial equivalence relations [24] and finitely-dimensional subspaces [7]. All these are examples of corelations and are treated in Section 4 below.

The case of props

As mentioned in the introduction, the motivating examples for our construction are categories providing a semantic interpretation for circuit diagrams. These are typically props (**pro**duct and **p**ermutation categories [17]): it is thus useful to phrase our construction in this setting.

Recall that a prop is a symmetric monoidal category with objects the natural numbers, in which $n \oplus m = n + m$. Props form a category **Prop** with morphisms the ioo strict symmetric monoidal functors. A simplification to Theorem 11 is that the coproduct $\mathcal{C} + \mathcal{C}'$ in **Prop** is computed as $\mathcal{C} +_{|\mathcal{C}|} \mathcal{C}'$ in **Cat**, because the set of objects is fixed for any prop.

For monoidal structure on $\mathcal{C}$ to extend to the categories of (co)spans and (co)relations, it is crucial that the monoidal product respects the ambient structure.

Let $(\mathcal{C}, \oplus)$ be a prop with pushouts, and let $(\mathcal{A}, \oplus)$ be a sub-prop. We say that **the monoidal product preserves pushouts** in $\mathcal{A}$ if, for all spans $N \leftarrow Y \rightarrow M$ and $N' \leftarrow Y' \rightarrow M'$ in $\mathcal{A}$, we have an isomorphism

$$(N \oplus N') +_{Y \oplus Y'} (M \oplus M') \cong (N +_Y M) \oplus (N' +_{Y'} M').$$

Note that this pushout is taken in $\mathcal{C}$. This condition holds, for example, whenever $\mathcal{C}$ is monoidally closed. We say the monoidal product preserves pullbacks if the analogous condition holds for pullbacks.

Furthermore, we say that a subcategory $\mathcal{A}$ is **closed under** $\oplus$ if, given morphisms f, g in $\mathcal{A}$, the morphism $f \oplus g$ is also in $\mathcal{A}$.

▶ **Theorem 18.** *Let $\mathcal{C}$ and $\mathcal{A}$ be props satisfying Assumption 10. Suppose that the monoidal product of $\mathcal{C}$ preserves pushouts in $\mathcal{C}$ and pullbacks in $\mathcal{A}$, and that $\mathcal{M}$ is closed under the monoidal product. Then we have a pushout square in* **Prop**

$$\begin{array}{ccc} \mathcal{A} + \mathcal{A}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{A}) \\ \downarrow & & \downarrow \Pi \\ \mathsf{Cospan}(\mathcal{C}) & \xrightarrow[\Gamma]{} & \mathsf{Corel}(\mathcal{C}). \end{array} \tag{10}$$

We also state the prop version of the abelian case. An abelian prop is just a prop which is also an abelian category and where the monoidal product is the biproduct.

▶ **Corollary 19.** Suppose that $\mathcal{C}$ is an abelian prop. The following is a pushout in **Prop**.

$$\begin{array}{ccc} \mathcal{C} + \mathcal{C}^{op} & \longrightarrow & \mathsf{Span}(\mathcal{C}) \\ \downarrow & & \downarrow \Pi \\ \mathsf{Cospan}(\mathcal{C}) & \xrightarrow[\Gamma]{} & \mathsf{Corel}(\mathcal{C}) \cong \mathsf{Rel}(\mathcal{C}) \end{array} \tag{$\triangle$}$$

Proofs of these results can be found in Appendix A.3 of the extended version of this paper.

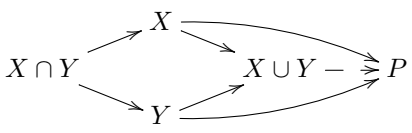
4 Examples

4.1 From Injections to Equivalence Relations

Our first example concerns the construction of equivalence relations starting from injective functions. For $n \in \mathbb{N}$, write $\overline{n}$ for the set $\{0, 1, \ldots, n\}$, and $\uplus$ for the disjoint union of sets. We fix a prop ER whose arrows $n \to m$ are the equivalence relations on $\overline{n} \uplus \overline{m}$. For composition $e_1 \,;\, e_2 \colon n \to m$ of equivalence relations $e_1 \colon n \to z$ and $e_2 \colon z \to m$, one first defines an equivalence relation on $\overline{n} \uplus \overline{z} \uplus \overline{m}$ by gluing together equivalence classes of e_1 and e_2 along common witnesses in $\overline{z}$, then obtains $e_1 \,;\, e_2$ by restricting to elements of $\overline{n} \uplus \overline{m}$.

Equivalence relations are equivalently described as corelations of functions. For this, let F be the prop whose arrows $n \to m$ are functions from $\overline{n}$ to $\overline{m}$. F has the usual factorisation system $(\mathsf{Su}, \mathsf{In})$ given by epi-mono factorisation, where Su and In are the sub-props of surjective and of injective functions respectively. Given these data, one can check that ER is isomorphic to $\mathsf{Corel}(\mathsf{F})$, the prop of corelations on F.

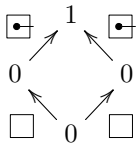
We are now in position to apply our construction of Theorem 18. First, we verify Assumption 10 with $\mathcal{C}$ instantiated as F and $\mathcal{A}$ as In. The only point requiring some work is the third, which goes as follows: given a cospan of monos $X \to P \leftarrow Y$, consider X, Y as subsets of P. Then the pullback-pushout diagram looks like



and $X \cup Y \to P$ is the inclusion map, hence a mono in In. Therefore, we can construct the pushout diagram (10) as follows:

$$\begin{array}{ccc} \mathsf{In} + \mathsf{In}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{In}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{F}) & \longrightarrow & \mathsf{ER} \end{array} \tag{11}$$

This modular reconstruction easily yields a presentation by generators and relations for (the arrows of) ER. Following the recipe of Corollary 17, ER is the quotient of $\mathsf{Cospan}(\mathsf{F})$ by all the equations generated by pullbacks in In, as in (8). Now, recall that In is presented (in string diagram notation [22]) by the generator $\boxdot : 0 \to 1$, and no equations. Thus, in order to present all the equations of shape (8) it suffices to consider a single pullback square in In:



, yielding the equation $\boxdot ; \boxdot \;=\; \square ; \square$. (12)

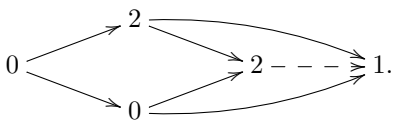
On the other hand, we know $\mathsf{Cospan}(\mathsf{F})$ is presented by the theory of special commutative Frobenius monoids (also termed separable Frobenius algebras), see [16]. Therefore ER is presented by the generators and equations of special commutative Frobenius monoids, with the addition of (12). This is known as the theory of extraspecial commutative Frobenius monoids [9]. This result also appears in [24], in both cases without the realisation that it stems from a more general construction.

4.2 From Functions to the Terminal Prop

It is instructive to see a non-example, to show that the assumptions on $\mathcal{A}$ are not redundant. One may want consider an obvious variation of (11), where instead of In one takes the whole F as $\mathcal{A}$. However, with this tweak the construction collapses: the pushout is the terminal prop $\mathbf{1}$ with exactly one arrow between any two objects.

$$\begin{array}{ccc} \mathsf{F} + \mathsf{F}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{F}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{F}) & \longrightarrow & \mathbf{1} \end{array} \tag{13}$$

This phenomenon was noted before ([7], see also [14, Th. 5.6]), however without an understanding of its relationship with other (non-collapsing) instances of the same construction. Theorem 18 explains why this case fails where others succeed: the problem lies in the choice of F as the subcategory $\mathcal{A}$. Indeed, the canonical map given by the pullback of the pushout of any span in $\mathcal{A} = \mathsf{F}$ does not necessarily lie in In, i.e. it may be not injective. An example is given by the cospan $0 \to 1 \leftarrow 2$, with the canonical map from the pushout of the pullback cospan the non-injective map $2 \to 1$:



4.3 From Injections to Partial Equivalence Relations

Partial equivalence relations (PERs) are common structures in program semantics, which date back to the seminal work of Scott [21] and recently revamped in the study of quantum computations (e.g., [15, 13]). Our approach yields a characterisation for the prop PER whose arrows $n \to m$ are PERs on $\overline{n} \uplus \overline{m}$, with composition as in ER. The ingredients of the construction generalise Example 4.1 from total to partial maps. Instead of F one starts with PF, the prop of partial functions, which has a factorisation system involving the sub-prop of partial surjections and the sub-prop of injections. The resulting prop of PF-corelations is isomorphic to PER. Theorem 18 yields the following pushout

$$\begin{array}{ccc} \mathsf{In} + \mathsf{In}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{In}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{PF}) & \longrightarrow & \mathsf{PER}. \end{array} \tag{14}$$

As in Example 4.1, following Corollary 17, (14) reduces the task of axiomatising PER to the one of axiomatising $\mathsf{Cospan}(\mathsf{PF})$ and adding the single equation (12) from $\mathsf{Span}(\mathsf{In})$. $\mathsf{Cospan}(\mathsf{PF})$ is presented by "partial" special commutative Frobenius monoids, studied in [24].

4.4 From Linear Maps to Subspaces

We now consider an example for the abelian case: the prop SV_{k} whose arrows $n \to m$ are k-linear subspaces of $\mathsf{k}^n \times \mathsf{k}^m$, for a field k. Composition in SV_{k} is relational: $V\,;W = \{(v,w) \mid \exists u.(v,u) \in V, (u,w) \in W\}$. Interest in SV_{k} is motivated by various recent applications. We mention the case where k is the field of Laurent series, in which SV_{k} constitutes a denotational semantics for signal flow graphs [3, 1, 5, 6], and the case $\mathsf{k} = \mathsf{Z}_2$, in which SV_{k} is isomorphic to the phase-free ZX-calculus, an algebra for quantum observables [8, 4].

Now, in order to apply our construction, note that SV_{k} is isomorphic to $\mathsf{Rel}(\mathsf{Vect}_k)$, where Vect_k is the abelian prop whose arrows $n \to m$ are the linear maps of type $\mathsf{k}^n \to \mathsf{k}^m$ (the monoidal product is by direct sum). This follows from the observation that subspaces of $\mathsf{k}^n \times \mathsf{k}^m$ of dimension z correspond to mono linear maps from k^z to $\mathsf{k}^n \times \mathsf{k}^m$, whence to jointly mono spans $n \leftarrow z \rightarrow m$ in Vect_k.

We are then in position to use Corollary 19, which yields the following pushout characterisation for SV_{k}.

$$\begin{array}{ccc} \mathsf{Vect}_k + {\mathsf{Vect}_k}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{Vect}_k) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{Vect}_k) & \longrightarrow & \mathsf{SV}_{\mathsf{k}}. \end{array} \tag{15}$$

This very same pushout has been studied in [4] for the $\mathsf{k} = \mathsf{Z}_2$ case. As before, the modular reconstruction suggests a presentation by generators and relations for SV_{k}, in terms of the theories for spans and cospans in Vect_k. The axiomatisation of SV_{k} is called the theory of interacting Hopf algebras [7, 23] , as it features two Hopf algebras structures and axioms expressing their combination.

On the top of existing results on SV_{k}, our Corollary 19 suggests a novel perspective, namely that SV_{k} can be also thought as the prop of *corelations* over $\mathsf{Vect}_{\mathsf{k}}$. This representation can be understood by recalling the 1-1 correspondence between subspaces of $\mathsf{k}^n \times \mathsf{k}^m$ and (solution sets of) homogeneous systems of equations $Mv = 0$, where M is a $z \times (n+m)$ matrix. Writing the block decomposition $M = (M_1 \mid -M_2)$, where M_1 is a $z \times n$ matrix and M_2 a $z \times m$ matrix, this is the same as solutions to $M_1 v_1 = M_2 v_2$. These systems then yield jointly epi cospans $n \xrightarrow{M_1} z \xleftarrow{M_2} m$ in $\mathsf{Vect}_{\mathsf{k}}$, that is, corelations.

4.5 From Free Module Homomorphisms to Linear Corelations

We now consider the generalisation of the linear case from fields to principal ideal domains (PIDs). In order to form a prop, we need to restrict our attention to finitely-dimensional *free* modules over a PID R. The symmetric monoidal category of such modules and module homomorphisms, with monoidal product by direct sum, is equivalent to the prop FMod_R whose arrows $n \to m$ are R-module homomorphisms $\mathsf{R}^n \to \mathsf{R}^m$ or, equivalently, $m \times n$-matrices in R. Because of the restriction to free modules, FMod_R is not abelian. However, it is still finitely bicomplete and has a costable (epi, split mono)-factorisation system.[4] Note that the fact that the ring R is a PID matters for the existence of pullbacks, as it is necessary for submodules of free R-modules to be free—pushouts exist by self-duality of FMod_R.

Write $\mathsf{MFMod}_\mathsf{R}$ for the prop of split monos in FMod_R. It is a classical, although nontrivial, theorem in control theory that this category obeys the required condition on pushouts of pullbacks [12]. Hence Theorem 18 yields the pushout square

$$\begin{array}{ccc} \mathsf{FMod}_\mathsf{R} + \mathsf{FMod}_\mathsf{R}{}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{MFMod}_\mathsf{R}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{FMod}_\mathsf{R}) & \longrightarrow & \mathsf{Corel}(\mathsf{FMod}_\mathsf{R}) \end{array} \tag{16}$$

in **Prop**. This modular account of $\mathsf{Corel}(\mathsf{FMod}_\mathsf{R})$ is relevant for the semantics of dynamical systems. When $\mathsf{R} = \mathbb{R}[s, s^{-1}]$, the ring of Laurent polynomials in some formal symbol s with coefficients in the reals, the prop $\mathsf{Corel}(\mathsf{FMod}_{\mathbb{R}[s,s^{-1}]})$ models complete linear time-invariant discrete-time dynamical systems in $\mathbb{R}$; more details can be found in [12]. In that paper, it is also proven that $\mathsf{Corel}(\mathsf{FMod}_\mathsf{R})$ is axiomatised by the presentation of $\mathsf{Cospan}(\mathsf{FMod}_\mathsf{R})$ with the addition of the law $\boxplus\,;\boxplus = \square$. By Corollary 17, it follows that $\boxplus\,;\boxplus = \square$ originates by a pullback in $\mathsf{Span}(\mathsf{MFMod}_\mathsf{R})$ and in this case it is the only contribution of spans to the presentation of corelations.

It is worth noticing that, even though FMod_R is not abelian, the pushout of spans and cospans over FMod_R does not have a trivial outcome as for the prop F of functions (Example 4.2). Instead, in [7, 23] it is proven that we have the pushout square

$$\begin{array}{ccc} \mathsf{FMod}_\mathsf{R} \oplus \mathsf{FMod}_\mathsf{R}{}^{op} & \longrightarrow & \mathsf{Span}(\mathsf{FMod}_\mathsf{R}) \\ \downarrow & & \downarrow \\ \mathsf{Cospan}(\mathsf{FMod}_\mathsf{R}) & \longrightarrow & \mathsf{SV}_\mathsf{k} \end{array} \tag{17}$$

in **Prop**, where k is the *field of fractions* of R.

The pushout (17) is relevant for the categorical semantics for signal flow graphs pursued in [3, 5, 6]. Even though it is not an instance of Theorem 18 or Corollary 19, our developments shed light on (17) through the comparison with (16). First, note that any element $r \in \mathsf{R}$ yields a module homomorphism $x \mapsto rx$ in FMod_R of type $1 \to 1$, represented as a string diagram $\boxed{r}$. The key observation is that, in (17), $\mathsf{Span}(\mathsf{FMod}_\mathsf{R})$ is contributing to the axiomatisation of SV_k (*cf.* Corollary 17) by adding, for each r, an equation

[4] The factorisation given by (epi, mono) morphisms is not unique up to isomorphism, whence the restriction to split monos—see [12].

in FMod_R. Back to (16), the only equations of this kind that $\mathsf{Span}(\mathsf{MFMod}_\mathsf{R})$ is contributing with are those in which  is a *split mono*, that means, when r is *invertible* in R. Therefore, the difference between (16) and (17) is that in the latter one is adding formal inverses  also for elements  which are not originally invertible in R. This explains the need of the field of fractions k of R in expanding the pushout object from $\mathsf{Corel}(\mathsf{FMod}_\mathsf{R})$ (in (16)) to $\mathsf{Corel}(\mathsf{Vect}_\mathsf{k}) \cong \mathsf{SV}_\mathsf{k}$ (in (17)).

4.6 From Maps of Algebras to Relations of Algebras

Let $\mathcal{C}$ be a regular category in which all regular epimorphisms split, and let T be a monad on $\mathcal{C}$. Then the Eilenberg–Moore category $\mathcal{C}^T$ is regular. As for any regular category, the (regular epi,mono)-factorisation system is stable, so we can construct the category $\mathsf{Rel}(\mathcal{C}^T)$ of relations in $\mathcal{C}^T$. With a few further conditions on T, we can apply Corollary 14 to realise $\mathsf{Rel}(\mathcal{C}^T)$ as a pushout of categories.

To see this, note that the regular epis in $\mathcal{C}^T$ are simply the algebra maps with underlying map in $\mathcal{C}$ a regular epi. Indeed, since by assumption regular epimorphisms in $\mathcal{C}$ split, coequalizers in $\mathcal{C}^T$ can be computed using coequalizers in $\mathcal{C}$. Moreover, as the forgetful functor $U\colon \mathcal{C}^T \to \mathcal{C}$ is monadic, it creates finite limits, and the canonical map from a span to the pullback of its pushout can also be computed in $\mathcal{C}$. Thus $\mathcal{C}^T$ satisfies Assumption 13 whenever it is finitely cocomplete and $\mathcal{C}$ satisfies Assumption 13. (Given the finite completeness of $\mathcal{C}$, it is in fact enough for $\mathcal{C}^T$ to have reflexive coequalizers: this implies finite cocompleteness.) This allows us to apply the construction of Corollary 14.

These conditions are met, for example, for any monad T over Vect_k. Hence, for example, we can apply Corollary 14 to the construction of the category of relations between algebras over a field (that is, vector spaces equipped with a bilinear product).

5 Concluding remarks

In summary, we have shown that categories of (co)relations may, under certain general conditions, be constructed as pushouts of categories of spans and cospans. In particular, especially since categories of spans and cospans can frequently be axiomatised using distributive laws, this offers a method of constructing axiomatisations of categories of (co)relations. Our results extend to the setting of props, and more generally symmetric monoidal categories. Moreover, these results are readily illustrated, unifying a diverse series of examples drawn from algebraic theories, program semantics, quantum computation, and control theory.

Looking forward, note that in the monoidal case the resulting (co)relation category is a so-named hypergraph category: each object is equipped with a special commutative Frobenius structure. Hypergraph categories are of increasing interest for modelling network-style diagrammatic languages, and recent work, such as that of decorated corelations [10] or the generalized relations of Marsden and Genovese [18], gives precise methods for tailoring constructions of these categories towards chosen applications. Our example on relations in categories of algebras for a monad (Subsection 4.6) hints at general methods for showing the present universal construction applies to these novel examples. We leave this as an avenue for future work.

References

1 John C. Baez and Jason Erbele. Categories in control. *Theory Appl. Categ.*, 30:836–881, 2015. URL: http://www.tac.mta.ca/tac/volumes/30/24/30-24abs.html.

2 John C. Baez and Brendan Fong. A compositional framework for passive linear circuits. *Preprint*, 2015. URL: https://arxiv.org/abs/1504.05625.

3 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. A categorical semantics of signal flow graphs. In *CONCUR 2014*, volume 8704 of *LNCS*, pages 435–450. Springer, 2014. doi:10.1007/978-3-662-44584-6_30.

4 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Interacting bialgebras are Frobenius. In *FoSSaCS 2014*, volume 8412 of *LNCS*, pages 351–365. Springer, 2014. doi:10.1007/978-3-642-54830-7_23.

5 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Full abstraction for signal flow graphs. In *POPL 2015*, pages 515–526, 2015. doi:10.1145/2676726.2676993.

6 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. The calculus of signal flow diagrams I: linear relations on streams. *Inf. Comput.*, 252:2–29, 2017. doi:10.1016/j.ic.2016.03.002.

7 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Interacting Hopf algebras. *Journal of Pure and Applied Algebra*, 221(1):144–184, 2017. doi:10.1016/j.jpaa.2016.06.002.

8 Bob Coecke and Ross Duncan. Interacting quantum observables: categorical algebra and diagrammatics. *New Journal of Physics*, 13(4):043016, 2011. doi:10.1088/1367-2630/13/4/043016.

9 Brandon Coya and Brendan Fong. Corelations are the prop for extraspecial commutative Frobenius monoids. *Theory Appl. Categ.*, 32(11):380–395, 2017. URL: http://www.tac.mta.ca/tac/volumes/32/11/32-11abs.html.

10 Brendan Fong. *The Algebra of Open and Interconnected Systems*. PhD thesis, University of Oxford, 2016. URL: https://arxiv.org/abs/1609.05382.

11 Brendan Fong. Decorated corelations. *Preprint*, 2017. URL: https://arxiv.org/abs/1703.09888.

12 Brendan Fong, Paolo Rapisarda, and Paweł Sobociński. A categorical approach to open and interconnected dynamical systems. In *LICS 2016*, pages 495–504, 2016. doi:10.1145/2933575.2934556.

13 Ichiro Hasuo and Naohiko Hoshino. Semantics of higher-order quantum computation via geometry of interaction. In *LICS 2011*, pages 237–246, 2011. doi:10.1109/LICS.2011.26.

14 Chris Heunen and Jamie Vicary. Lectures on categorical quantum mechanics, 2012.

15 Bart Jacobs and Jorik Mandemaker. Coreflections in algebraic quantum logic. *Foundations of Physics*, 42(7):932–958, 2012. doi:10.1007/s10701-012-9654-8.

16 Stephen Lack. Composing PROPs. *Theory Appl. Categ.*, 13(9):147–163, 2004. URL: http://www.tac.mta.ca/tac/volumes/13/9/13-09abs.html.

17 Saunders Mac Lane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71:40–106, 1965. doi:10.1090/S0002-9904-1965-11234-4.

18 Dan Marsden and Fabrizio Genovese. Custom hypergraph categories via generalized relations. *Preprint*, 2017. URL: https://arxiv.org/abs/1703.01204.

19 Jeanne Meisen. On bicategories of relations and pullback spans. *Communications in Algebra*, 1(5):377–401, 1974. doi:10.1080/00927877408548625.

20 Robert Rosebrugh and R.J. Wood. Distributive laws and factorization. *Journal of Pure and Applied Algebra*, 175(1–3):327 – 353, 2002. doi:10.1016/S0022-4049(02)00140-8.

21 Dana Scott. Data types as lattices. *SIAM Journal on Computing*, 5(3):522–587, 1976. doi:10.1137/0205037.

22 Peter Selinger. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pages 289–355. Springer, 2011. doi:10.1007/978-3-642-12821-9_4.

23 Fabio Zanasi. *Interacting Hopf Algebras: the theory of linear systems*. PhD thesis, Ecole Normale Supérieure de Lyon, 2015.

24 Fabio Zanasi. The algebra of partial equivalence relations. *Electr. Notes Theor. Comput. Sci.*, 325:313–333, 2016. doi:10.1016/j.entcs.2016.09.046.

Sequoidal Categories and Transfinite Games: A Coalgebraic Approach to Stateful Objects in Game Semantics*†

William John Gowers[1] and James Laird[2]

1 **Department of Computer Science, University of Bath, Claverton Down, Bath, BA2 7AY, United Kingdom**
W.J.Gowers@bath.ac.uk

2 **Department of Computer Science, University of Bath, Claverton Down, Bath, BA2 7AY, United Kingdom**
jiml@cs.bath.ac.uk

Abstract

The non-commutative sequoid operator $\oslash$ on games was introduced to capture algebraically the presence of state in history-sensitive strategies in game semantics, by imposing a causality relation on the tensor product of games. Coalgebras for the functor $A \oslash \_$ – i.e., morphisms from S to $A \oslash S$ – may be viewed as state transformers: if $A \oslash \_$ has a *final coalgebra*, $!A$, then the anamorphism of such a state transformer encapsulates its explicit state, so that it is shared only between successive invocations.

We study the conditions under which a final coalgebra $!A$ for $A \oslash \_$ is the carrier of a *cofree commutative comonoid* on A. That is, it is a model of the exponential of linear logic in which we can construct imperative objects such as reference cells coalgebraically, in a game semantics setting. We show that if the tensor *decomposes* into the sequoid, the final coalgebra $!A$ may be endowed with the structure of the cofree commutative comonoid if the natural isomorphism $!(A \times B) \cong !A \otimes !B$ holds. This condition is always satisfied if $!A$ is the *bifree algebra* for $A \oslash \_$, but in general it is necessary to impose it, as we establish by giving an example of a sequoidally decomposable category of games in which plays will be allowed to have transfinite length. In this category, the final coalgebra for the functor $A \oslash \_$ is not the cofree commutative comonoid over A: we illustrate this by explicitly contrasting the final sequence for the functor $A \oslash \_$ with the chain of symmetric tensor powers used in the construction of the cofree commutative comonoid as a limit by Melliès, Tabareau and Tasson [23].

1998 ACM Subject Classification F3.2.2 Denotational Semantics

Keywords and phrases Game semantics, Stateful languages, Transfinite games, Sequoid operator

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.13

1 Introduction

Game semantics has been used to define a variety of models of higher-order programming languages with mutable state, including Idealized Algol [2], and various fragments of ML [3, 4]. Unlike traditional denotational semantics, which typically represent imperative programs as state transformers, the state in these models is completely implicit: local

* This work was partially supported by UK EPSRC Grant EP/K037633/1.
† Full version with appendix available at [10], https://arxiv.org/abs/1706.00035.

declaration of mutable variables is interpreted as composition with a "history sensitive" strategy representing a reference cell. This is conceptually simple in principle but leads to some quite combinatorial definitions; a more explicit representation of the current state can be very useful for constructing and reasoning about imperative objects.

1.1 Defining Higher-order Stateful Objects, Coalgebraically

Let us first motivate the study of the coalgebraically derived cofree comonoid in game semantics by considering a similar but simpler and more familiar phenomenon. A *state-transformer* in a symmetric monoidal category is a morphism $f : A \otimes S \to B \otimes S$ taking an argument together with an input state to a result together with an output state. A well-studied [13] technique in semantics is to use an appropriate final coalgebra to *encapsulate* the state in such a transformer, allowing multiple successive invocations, each of which passes its output state as an input state to the next invocation.

For example, consider the category Rel of sets and relations, with symmetric monoidal structure given by the Cartesian product (with unit I, the singleton set $\{*\}$). This has finite (bi)products (disjoint unions) so we may define the functor $F(A, S) = (A \otimes S) \oplus I$. For any object (set) A, let A^* be the set of finite sequences of elements of A (i.e. the carrier of the free monoid on A), and $\alpha : A^* \to F(A, A^*)$ be the morphism $\{(\varepsilon, \mathsf{inr}(*)\} \cup \{(aw, (\mathsf{inl}(a, w)) \mid a \in A, w \in A^*\}$. It is straightforward to show that:

▶ **Lemma 1.** *(A^*, α_A) is the final coalgebra for $F(A, \_)$.*

Since we have a natural transformation $\mathsf{inl}_{A,S} : A \otimes S \to F(A, S)$, we may encapsulate the state in the state transformer $f : S \to A \otimes S$ by taking the *anamorphism* of $\tilde{f} = (f; \mathsf{inl}_{A,S}) \cup \{(s, \mathsf{inr}(*)) : s \in S\} : S \to F(A, S)$, – i.e. the unique $F(A, \_)$-coalgebra morphism from $(S, \tilde{f})$ into (A^*, α_A). This is a morphism from an initial state S into A^*: by definition, composing it with $\alpha : A \to F(A, A^*)$ (which we can think of as *invoking* our stateful object) returns a copy of f together with the encapsulated morphism with updated internal state.

Distributivity of $\oplus$ over $\otimes$ implies that $F(A \oplus A', S) \cong F(A, S) \oplus F(A', S)$. This allows state transformers to be aggregated, to construct stateful objects compounded of a series of methods which share access to a common state. For example, we may represent a reference cell storing integer values as a state transformer $\mathsf{cell} : \mathbb{N} \to (\mathbb{N} \oplus \mathbb{N}) \otimes \mathbb{N}$, obtained by aggregating two "methods" which share access to a value in $\mathbb{N}$ representing the contents of the cell – returning a "read" of the input state (and leaving it unchanged) or accepting a "write" of a new value and using it to update the state. Thus (with appropriate tagging) it is the relation $\{(i, (\mathtt{read}(i), i)) : i \in \mathbb{N}\} \cup \{(i, (\mathtt{write}(j), j)) : i, j \in \mathbb{N}\}$. The anamorphism of the coalgebra $\tilde{\mathsf{cell}} : \mathbb{N} \to F(\mathbb{N} \oplus \mathbb{N}, \mathbb{N})$ is the relation from $\mathbb{N}$ to $(\mathbb{N} \oplus \mathbb{N})^*$ consisting of pairs of the form $(i_1, \mathtt{read}(i_1)^*\mathtt{write}(i_2)\mathtt{read}(i_2)^* \ldots)$. Composition with this morphism is precisely the interpretation of new variable declaration in the semantics in Rel of the prototypical functional-imperative language *Syntactic Control of Interference* (SCI) given in [22].

Coalgebraic methods thus give us a recipe for constructing and using categorical definitions of stateful semantic objects. In order to fully exploit these, however, we endow A^* with the structure of a *comonoid* in our symmetric monoidal category, by defining morphisms $\delta_A : A^* \to A^* \otimes A^* = \{(u \cdot v, (u, v)) \mid u, v \in A^*\}$ and $\epsilon : A \to I = \{(\varepsilon, *)\}$. In fact, this is the *cofree comonoid* on A – there is a morphism $\eta_A : A^* \to A = \{(a, a) \mid a \in A\}$ such that for any comonoid $(B, \delta_B, \epsilon_B)$, composition with η_A defines an equivalence (natural in B) between the morphisms from B into A, and the comonoid morphisms from $(B, \delta_B, \epsilon_B)$ into $(A^*, \delta_A, \epsilon_A)$.

▶ **Proposition 2.** *(A^*, δ, ϵ) is the cofree comonoid on* Rel.[1]

This structure can be used to interpret procedures which share access to a stateful resource such as a reference cell. Its main limitation is that we have not defined a *commutative* comonoid for any non-empty set A (evidently, δ is not invariant under post-composition with the symmetry isomorphism of the tensor). Thus we can only model procedures with shared access to the same stateful object if the order in which they are permitted to access it is fixed. (This is precisely the situation in SCI, where the typing system allows sharing across sequential composition, but not between functions and their arguments.) In order to model sharing of state without this constraint (and build a Cartesian closed category), we need to endow our final coalgebra with the structure of a *cofree commutative comonoid*, proposed as the basis of a model of linear logic by Lafont [15]. The category of sets and relations does not allow this (the cofree commutative comonoid on an object A in Rel is given by the set of finite multisets of A, which is not a final coalgebra). Hence, we turn to the richer structures of game semantics.

1.2 The cofree commutative comonoid as a final coalgebra

We now outline the remainder of the paper. Our main contribution is an investigation of the circumstances in which the cofree commutative comonoid on A arises from a final coalgebra for the functor $A \oslash \_$, where $\oslash$ (the sequoid) is a non-commutative operation on games introduced by one of the authors [16].[2] In this setting, we can model a state transformer for a program as a morphism $S \to A \oslash S$ – i.e., a coalgebra for the functor $A \oslash \_$. The final coalgebra for this functor is the exponential game $!A$ introduced by Hyland [11], which corresponds to a ω-fold sequence $A \oslash (A \oslash (A \oslash \ldots))$; under appropriate conditions, it is the carrier for the cofree commutative comonoid on A. We aim to characterize these conditions using just the categorical structure, in order to capture a general class of models and to derive formal principles for coinductively proving program equivalences. In a nutshell, we require that a certain natural morphism $!A \otimes !B \to !(A \times B)$ is an isomorphism. This can be used to show that $!\_$ gives rise to a strong monoidal functor. Perhaps more surprisingly, this is sufficient to show that $!A$ is the cofree commutative comonoid.

This *strong monoidal hypothesis* holds whenever $!A$ is a bifree algebra for $A \oslash \_$. But we are also interested in cases where $!A$ is not bifree – for example, in categories of "win games" and *winning strategies* [11], which lack the partial maps which can be shown to arise in the bifree case. To show that the strong monoidal hypothesis is necessary in general, we introduce a sequoidal category of games with *transfinite* plays in which it does not hold: because a transfinite interleaving of two sequences of length ω may have length greater than ω, the final coalgebra for $A \oslash \_$ (corresponding to only ω-many copies of the game A) cannot be the carrier for the cofree commutative comonoid.

We compare the coalgebraic construction of the cofree exponential to the explicit characterization of the latter given by Melliès, Tabareau and Tasson [23] as the limit of a chain of symmetric tensor powers. This chain exists in any decomposable sequoidal category: where its limit exists and is preserved by the tensor (the conditions required in [23]) it must be the

[1] The definitions of δ and ϵ, and the proof that this is the cofree comonoid may be derived from the fact that (A^*, α_A) is a *bifree algebra* for $F(\_, A)$ – i.e. (A^*, α^{-1}) is an initial algebra for $F(A, \_)$ (α must be an isomorphism by Lambek's lemma). We leave this as an exercise.

[2] We will focus on a particular category of "history sensitive", Abramsky-Jagadeesan style games [1], but sequoidal structure is a unifying feature of sequential, history-sensitive games: see [16] for a variant of the Hyland-Ong games and [9] for Conway games.

final coalgebra for $A \oslash \_$. However, in our categories of transfinite games, and win games and winning strategies (which may be viewed as games of length $\omega + 1$), the construction fails – this limit is not the cofree commutative comonoid.

2 Sequoidal categories

2.1 Game semantics and the sequoidal operator

We shall present a form of game semantics in the style of [11] and [1]. A game A is given by a set M_A of *P-moves* and *O-moves* and by a non-empty prefix-closed set $P_A \subseteq M_A^*$ of *positions*, which are alternating sequences of O-moves and P-moves. We shall adopt the rule that all positions must start with an O-move. We call a position a *P-position* if it ends with a P-move or is empty and an *O-position* if it ends with an O-move.

A *strategy* for a game A is a non-empty prefix-closed subset σ of P_A that is closed under O-replies to P-positions and which satisfies *determinism*: if $sa, sb \in \sigma$, where s is an O-position, then $a = b$.

We build connectives on games as in [1]. The set of moves for a compound game is given by the disjoint union of the sets of moves for the individual sub-games, and the positions for each game are defined as follows:

Product If $(A_i : i \in I)$ is a collection of games, then we write $\prod_{i\in I} A_i$ for the game in which player O, on his first move, may play in any of the games A_i. From then on, play continues in A_i. If A_1, A_2 are games, we write $A_1 \times A_2$ for $\prod_{i=1}^{2} A_i$.

Tensor Product If A, B are games,the tensor product $A \otimes B$ is played by playing the games A and B in parallel, where player O may elect to switch games whenever it is his turn and continue play in the game he has switched to.

Linear implication The implication $A \multimap B$ is played by playing the game B in parallel with the *negation* of A - that is, the game formed by switching the roles of players P and O in A. Since play in the negation of A starts with a P-move, player O is forced to make his first move in the game B. Thereafter, player P may switch games whenever it is her turn.

It is well known (see [1], for example) that we may compose strategies σ for $A \multimap B$ and τ for $B \multimap C$ to get a morphism $\sigma;\tau$ for $A \multimap C$ and that this structure gives rise to a monoidal closed category where objects are games, morphisms from A to B are strategies for $A \multimap B$ and the tensor product and linear implication are given by $A \otimes B$ and $A \multimap B$. We call this category $\mathcal{G}$. $\mathcal{G}$ has all products, given by $\prod_{i\in I} A_i$ as above.

The one non-standard connective we will use is the *sequoid* connective from [16]:

Sequoid If A and B are games, then the positions of $A \oslash B$ are precisely the positions of $A \otimes B$ that are empty or that start with a move in A.

By inspection, we can verify that we have structural isomorphisms:

$$\texttt{dist}\colon A \otimes B \xrightarrow{\cong} (A \oslash B) \times (B \oslash A) \qquad \texttt{dist}^0\colon I \oslash C \xrightarrow{\cong} I$$
$$\texttt{dec}\colon (A \times B) \oslash C \xrightarrow{\cong} (A \oslash C) \times (B \oslash C) \qquad \texttt{r}\colon A \oslash I \xrightarrow{\cong} A$$
$$\texttt{passoc}\colon (A \oslash B) \oslash C \xrightarrow{\cong} A \oslash (B \otimes C)$$

We might expect that the sequoid would give rise to a functor $\mathcal{G} \times \mathcal{G} \to \mathcal{G}$ in the way that the tensor product does, through playing strategies in parallel. However, this does not quite work: playing strategies σ for $A \multimap B$ and τ for $C \multimap D$ in parallel does not necessarily give rise to a valid strategy for $(A \oslash C) \multimap (B \oslash D)$, since player P might end up playing in C before anyone has played in A. However, if we require that the strategy σ is *strict* – that is,

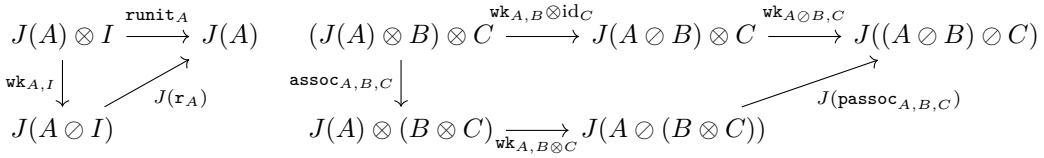
that player P's reply (if any) to the opening move in B is always a move in A – then we do get a valid strategy $\sigma \oslash \tau$ for $(A \oslash C) \multimap (B \oslash D)$ and, moreover, $\sigma \oslash \tau$ is strict. We shall write $\mathcal{G}_s$ for the category of games with *strict* strategies as morphisms; then $\_ \oslash \_$ gives us a functor $\mathcal{G}_s \times \mathcal{G} \to \mathcal{G}_s$.

2.2 Sequoidal categories

We now formalize these observations into a category-theoretic definition. The purpose of this definition is to formalize precisely what it is about categories of games that makes them suitable for modeling stateful programs, and to give an equational characterization of the combinatorial definitions used in the Abramsky-McCusker model of Idealized Algol [16, 2].

▶ **Definition 3.** A *sequoidal category* consists of the following data:

- A symmetric monoidal category $\mathcal{C}$ with monoidal product $\otimes$ and tensor unit I, associators $\mathtt{assoc}_{A,B,C} \colon (A \otimes B) \otimes C \xrightarrow{\cong} A \otimes (B \otimes C)$, unitors $\mathtt{runit}_A \colon A \otimes I \xrightarrow{\cong} A$ and $\mathtt{lunit}_A \colon I \otimes A \xrightarrow{\cong} A$ and braiding $\mathtt{sym}_{A,B} \colon A \otimes B \to B \otimes A$.
- A category $\mathcal{C}_s$.
- A right monoidal category action [14] of $\mathcal{C}$ on the category $\mathcal{C}_s$. That is, a functor $\_ \oslash \_ : \mathcal{C}_s \times \mathcal{C} \to \mathcal{C}_s$ that gives rise to a monoidal functor from $\mathcal{C}$ into the category of endofunctors on $\mathcal{C}_s$. We write $\mathtt{passoc}_{A,B,C} \colon (A \oslash B) \oslash C \to A \oslash (B \otimes C)$ and $\mathtt{r}_A \colon A \oslash I \to A$ for the coherence parts of this monoidal functor.
- A functor $J \colon \mathcal{C}_s \to \mathcal{C}$ (in the games example, this is the inclusion functor $\mathcal{G}_s \to \mathcal{G}$)
- A natural transformation $\mathtt{wk}_{A,B} \colon J(A) \otimes B \to J(A \oslash B)$ satisfying the coherence conditions[3]:



Our category of games satisfies further conditions:

▶ **Definition 4.** Let $\mathcal{C} = (\mathcal{C}, \mathcal{C}_s, J, \mathtt{wk})$ be a sequoidal category. We say that $\mathcal{C}$ is an *inclusive sequoidal category* if $\mathcal{C}_s$ is a full-on-objects subcategory of $\mathcal{C}$ containing all isomorphisms and finite products of $\mathcal{C}$, and the morphisms $\mathtt{wk}_{A,B}$ and J is the inclusion functor.

We say that $\mathcal{C}$ is *decomposable* if I is a terminal object for $\mathcal{C}$ and if for any A and B, the tensor product $A \otimes B$ is a Cartesian product of $A \oslash B$ and $B \oslash A$, with projections $\mathtt{wk}_{A,B} : A \otimes B \to A \oslash B$ and $\mathtt{sym}_{A,B}; \mathtt{wk}_{A,B} : A \otimes B \to B \oslash A$. We say that $\mathcal{C}$ is *distributive* if whenever the product $\prod_{i \in I} A_i$ exists, then $\left(\prod_{i \in I} A_i\right) \oslash B$ is the product of the $A_i \oslash B$, with projections $\mathrm{pr}_i \oslash \mathrm{id}_B$, and if it has a terminal object 1 satisfying $1 \oslash A \cong 1$ for all objects A.

Although there are important examples where we do not have products, our examples will all be categories with all products. Then we can state the definitions of decomposability

[3] These coherence conditions say that $(J, \mathtt{wk})$ is a *lax morphism of right monoidal actions of* $\mathcal{C}$ from the sequoidal action $(\mathcal{C}_s, \_ \oslash \_)$ to the 'right multiplication' action $(\mathcal{C}, \_ \otimes \_)$.

and distributivity more succinctly by requiring that the natural transformations

$$\mathtt{dec}_{A,B} = \langle \mathtt{wk}_{A,B}, \mathtt{sym}_{A,B}; \mathtt{wk}_{A,B} \rangle \colon A \otimes B \to (A \oslash B) \times (B \oslash A)$$

$$\mathtt{dec}^0 \colon I \to 1$$

$$\mathtt{dist}_{A,B,C} = \langle \mathrm{pr}_1 \oslash \mathrm{id}_C, \mathrm{pr}_2 \oslash \mathrm{id}_C \rangle \colon (A \times B) \oslash C \to (A \oslash C) \times (B \oslash C)$$

$$\mathtt{dist}_{(A_i \,:\, i \in I),B} = \langle \mathrm{pr}_i \oslash \mathrm{id}_C \,:\, i \in I \rangle \colon \left(\prod_{i \in I} A_i \right) \oslash C \to \prod_{i \in I} (A_i \oslash C)$$

$$\mathtt{dist}_{A,0} \colon 1 \oslash A \to 1$$

are isomorphisms.

The category of games $\mathcal{G}$ and categories arising in different traditions of game semantics [9, 16] are the prototype examples of distributive, decomposable sequoidal categories.

▶ Remark. Churchill, Laird and McCusker give a result in [8] that implies that any sequoidal category satisfying these, and other, extra conditions can be used to model a proof calculus for describing games and strategies. Nevertheless, they note that examples do exist of sequoidal categories that do not arise as categories of games, such as the category of locally Boolean domains described in [17] (see the last paragraph of that paper, and also the citation in [8]).

2.3 The sequoidal exponential

There are several ways to add exponentials to the basic category of games, but the definition that fits our purposes is the one based on countably many copies of the base game (see [11], for example): the exponential $!A$ of A is the game in which player O may switch between countably many copies of A – $A_0, A_1, A_2, \ldots$, as long as he starts them in order, starting with A_0, then opening A_1 and so on. This condition on the order in which games may be opened is very important, as it allows us to define the exponential morphisms $!A \to !A \otimes !A$ and $!A \to !!A$. In the first case, it can be proved [19] that the comultiplication $!A \to !A \otimes !A$ exhibits $!A$ as the *cofree commutative comonoid* on A, which shows that A is a suitable model for the exponential [15].

Even more interestingly from the point of view of modelling stateful languages, we may characterize $!A$ as the *final coalgebra* for the functor $J(A \oslash \_) \colon \mathcal{G} \to \mathcal{G}$ (henceforth we shall write this functor as $A \oslash \_$, eliding the inclusion functor J). That is, given a coalgebra from $\oslash\_$ – a game B and a morphism $\sigma \colon B \to A \oslash B$ – we get a unique morphism $(\!|\, \sigma \,|\!)$ making the following diagram commute:

$$\begin{array}{ccc} B & \xrightarrow{\sigma} & A \oslash B \\ {\scriptstyle (\!|\, \sigma \,|\!)}\downarrow & & \downarrow{\scriptstyle \mathrm{id}_A \oslash (\!|\, \sigma \,|\!)} \\ !A & \xrightarrow[\alpha]{} & A \oslash !A \end{array}$$

We call $(\!|\, \sigma \,|\!)$ the *anamorphism* of σ.

We shall use the following standard pieces of coalgebra theory:

Lambek's Lemma α_A is an isomorphism, with inverse given by the anamorphism of the map $\mathrm{id} \oslash \alpha_A \colon A \oslash !A \to A \oslash (A \oslash !A)$ [20]. In particular, α_A is a morphism in $\mathcal{G}_s$. In the general case, we deduce that α_A is a morphism in $\mathcal{C}_s$.

Final Sequence If $\mathcal{C}$ is a category with enough limits and F is an endofunctor on $\mathcal{C}$, we may build up an ordinal indexed sequence of objects and morphisms of $\mathcal{C}$ (that is, a functor $\mathsf{Ord}^{\mathrm{op}} \to \mathcal{C}$, where Ord is the category of ordinals and prefix inclusions):

$$1 \leftarrow F(1) \leftarrow F^2(1) \leftarrow \cdots F^\omega(1) \leftarrow F^{\omega+1}(1) \leftarrow \cdots$$

(by repeatedly applying F and taking limits). If this sequence stabilizes for any δ (i.e., if the morphism from $F^{\delta+1}(1) \to F^{\delta}(1)$ is an isomorphism), then $F^{\delta}(1)$ is the final coalgebra for F [27]. In the case $F = A \oslash \_$, we shall write $A^{\oslash\delta}$ for $F^{\delta}(1)$.

2.4 Imperative programs as anamorphisms

We now illustrate the construction of stateful objects using anamorphisms by constructing the strategy cell from [2] that represents a storage cell. If X is a set of values, write $\underline{X}$ for the game denoting the corresponding type: that is, X is the game with maximal plays qx, where x ranges over the elements of X, and $\underline{X}$ has canonical strategies $\underline{x}$ for $x \in X$, in which player P responds to the opening move q with the move x. In particular, the game $\underline{\{*\}}$ corresponding to a singleton set denotes the void or command type com. We shall write $\Sigma = \underline{\{*\}}$ for this game, and write $\mathtt{OK} = \underline{*}$ for its unique total strategy.

Following [2], we define $\mathtt{Var}[X]$ to be the type $\mathsf{com}^X \times X$; that is, the product of X-many copies of the command type with one copy of the type X. We can think of this with an object that has a method $\mathtt{write\_}x$ for each element x, together with a method $\mathtt{read}$. The corresponding game is the game

$$\mathtt{Var}[X] = \Sigma^X \times \underline{X}$$

In order to tell apart the various games, we shall write Σ_x for each copy of Σ, and write the moves of Σ_x as q_x and $*_x$. Let $d \in X$ be a fixed default value. Then it is quite easy to describe what the strategy cell on $!\mathtt{Var}[X]$ should be: it is the strategy that always responds to q_x with $*_x$ (as it is forced to do) and which responds to the move q in $\underline{X}$ with that value $x \in X$ such that q_x has been played most recently (or with d if player O has not yet played in Σ^X). This is more or less how the strategy is defined in [2]. The problem is that the state (the current most recently written value of x) is *implicit*, and it is hard to get a handle on it.

Instead, we try a state-transformer based approach. We shall use $!\underline{X}$ to represent the state of the storage cell (the ! is there since we will need to refer to the state multiple times). We define morphisms $\mathtt{read} \colon !\underline{X} \to \underline{X} \oslash !\underline{X}$ and $\mathtt{write\_}x \colon !\underline{X} \to \Sigma \times !\underline{X}$ as follows: $\mathtt{read}$ is the canonical morphism $\alpha_{\underline{X}}$, while $\mathtt{write\_}x$ is the following composite, which throws away the previous state and updates it with the value x:

$$!\underline{X} \xrightarrow{\mathsf{weak}} I \xrightarrow{\mathtt{runit}} I \otimes I \xrightarrow{\mathtt{OK}\otimes !\underline{x}} \Sigma \otimes !\underline{X} \xrightarrow{\mathtt{wk}} \Sigma \oslash !\underline{X}$$

Taking the product of these morphisms and applying the distributivity of $\times$ over $\oslash$, we get our state transformer:

$$\mathsf{cell\_ST} \colon !\underline{X} \xrightarrow{\langle \mathtt{write}_x \colon x \in X \,,\, \mathtt{read} \rangle} (\Sigma \oslash !\underline{X})^X \times \underline{X} \oslash !\underline{X} \xrightarrow{\mathtt{dist}^{-1}} (\Sigma^X \times \underline{X}) \oslash !\underline{X}$$

By the definition of $!\mathtt{Var}[X]$, there is now a unique morphism $\mathsf{cell\_init} \colon !X \to !\mathtt{Var}[X]$ making the following diagram commute:

$$\begin{array}{ccc} !\underline{X} & \xrightarrow{\mathsf{cell\_ST}} & \mathtt{Var}[X] \oslash !\underline{X} \\ {\scriptstyle \mathsf{cell\_init}}\downarrow & & \downarrow{\scriptstyle \mathrm{id} \oslash \mathsf{cell\_init}} \\ !\mathtt{Var}[X] & \xrightarrow{\alpha} & \mathtt{Var}[X] \oslash !\mathtt{Var}[X] \end{array}$$

Define a strategy $\sigma \colon !\underline{X} \to !\mathtt{Var}[X]$ combinatorially by saying that σ is the strategy that behaves like cell (as defined above) on $!\mathtt{Var}[X]$ but which interrogates its argument in order to establish the default value, rather than using a fixed value. By inspection, we can verify that

replacing cell_init with σ in the diagram above makes the square commute, and therefore that cell_init $= \sigma$ by uniqueness. It follows that cell is equal to the following composite:

$$I = !I \xrightarrow{!d} !X \xrightarrow{\text{cell\_init}} !\mathtt{Var}[X]$$

But now we are able to reason about its state explicitly by using the coalgebraic definition.

▶ **Exercise 1.** *By modifying the definition of* `write_x`*, show that the same construction may be used to model a stack with* push *and* pop *methods.*

3 Constructing cofree commutative comonoids in sequoidal categories

3.1 A formula for the sequoidal exponential

We observed that the exponential $!A$ of a game A arises as the final coalgebra for the functor $A \oslash \_$. We also observed that $!A$ has the structure of a cofree commutative comonoid on A. These two facts are both crucial if we want to use sequoidal categories to model stateful programs. In this section, we shall consider conditions under which we may deduce that the final coalgebra for $A \oslash \_$ and the cofree commutative comonoid over A coincide.

One important result is the formula given by Melliès, Tabareau and Tasson [23], which does not depend on the presence of Cartesian products but which obtains the cofree commutative comonoid as a limit of *symmetric tensor powers*.

▶ **Definition 5.** If A is an object in a symmetric monoidal category, a n-fold symmetric tensor power of A is an *equalizer* (A^n, eq) for the group G of symmetry automorphisms on $A^{\otimes n}$. A tensor power is preserved by the tensor product if $(B \otimes A^n, \mathrm{id}_B \otimes \mathsf{eq})$ is an equalizer for the automorphisms $\{\mathrm{id}_B \otimes g \mid g \in G\}$.

In any affine category[4] with symmetrized tensor powers of A we may define a diagram $\Delta(A) =$

$$I \xleftarrow{p_0} A \xleftarrow{p_1} A^2 \xleftarrow{p_2} \cdots \xleftarrow{p_{i-1}} A^i \xleftarrow{p_i} \cdots$$

where $p_i : A^{i+1} \to A^i$ is the unique morphism given by the universal property of the symmetric tensor power, such that $p_i ; \mathsf{eq}_i : A^{i+1} \to A^{\otimes i} = \mathsf{eq}_{i+1} ; (A^{\otimes i} \otimes t_A)$.

Melliès, Tabareau and Tasson [23] have shown that where the limit $(A^\infty, \{p_i^\infty : A^\infty \to A^i\})$ for this diagram exists and commutes with the tensor, – i.e. for each object B, $B \otimes A^\infty$ is the limit of

$$B \otimes I \xleftarrow{\mathrm{id}_B \otimes p_0} B \otimes A \xleftarrow{\mathrm{id}_B \otimes p_1} B \otimes A^2 \xleftarrow{\mathrm{id}_B \otimes p_2} \cdots$$

then a comultiplication $\mu : A^\infty \to A^\infty \otimes A^\infty$ may be defined making $(A^\infty, \mu, t_{!A})$ the cofree commutative comonoid. Where these conditions are satisfied, we shall call this a MTT-exponential.

In the category of games, the morphisms $\mathrm{id} \oslash \mathrm{pr}_1 : A \oslash (B \times C) \to A \oslash B$ and $\mathrm{id} \oslash \mathrm{pr}_2 : A \oslash (B \times C) \to A \oslash C$ are jointly monomorphic, and this joint monomorphism is preserved by the tensor product. If a distributive sequoidal category satisfies the same property, we say that it is *strong distributive*.

[4] This is a special case of the situation considered in [23]: that A is a "free pointed object".

▶ **Proposition 6.** *Any strong distributive decomposable sequoidal category has all symmetric tensor powers, and these are preserved by the tensor.*

Proof. By sequoidal decomposability, for any $n \in \mathbb{N}$, $A^{\otimes(n+1)}$ is the Cartesian product $\prod_{i \leq n}(\mathrm{id}_A \oslash A^{\otimes n})$ with projections $\mathtt{sym}_i; \mathtt{wk}_{A,A^{\otimes n}}$, where $\mathtt{sym}_i : A^{\otimes(n+1)} \to A^{\otimes(n+1)}$ is the symmetry isomorphism corresponding to the permutation on n which swaps 1 and i.

Define $\mathtt{wk}^n : A^{\otimes n} \to A^{\oslash n}$ by $\mathtt{wk}^{n+1} = \mathtt{wk}_{A,A^{\otimes n}}; (\mathrm{id}_A \oslash \mathtt{wk}^n)$. We show (by induction on n) that for any n, the morphisms $\mathtt{sym}^\pi; \mathtt{wk}_n$ are jointly monomorphic, where $\mathtt{sym}^\pi$ ranges over all of the permutation isomorphisms on $A^{\otimes n}$, and that this joint monomorphism is preserved by the tensor product.

We define the equalizer $\mathtt{eq}_n : A^{\oslash n} \to A^{\otimes n}$ inductively by setting $\mathtt{eq}_n$ to be the product $\langle \mathrm{id}_A \oslash \mathtt{eq}_{n-1}, \ldots, \mathrm{id}_A \oslash \mathtt{eq}_{n-1} \rangle$, using the identification of $A^{\oslash n}$ as a product given above. We may show inductively that $\mathtt{eq}_n; \mathtt{sym}^\pi; \mathtt{wk}^n = \mathrm{id}$ for all permutations $\pi \in S_n$. Given any $f : C \to A^{\otimes n} \otimes B$ such that $f; (\mathtt{sym}^\pi \otimes \mathrm{id}_B) = f$ for any permutation π, taking $f; (\mathtt{wk}^n \otimes \mathrm{id}_B) : C \to A^{\oslash n} \otimes B$ gives the unique morphism such that $f; (\mathtt{wk}^n \otimes \mathrm{id}_B); (\mathtt{eq}_n \otimes \mathrm{id}_B) = f$. Indeed, for all permutations $\pi \in S_n$ we have $f; ((\mathtt{wk}^n; \mathtt{eq}_n; \mathtt{sym}^\pi; \mathtt{wk}_n) \otimes \mathrm{id}_B) = f; (\mathtt{wk}^n \otimes \mathrm{id}_B) = f; ((\mathtt{sym}^\pi; \mathtt{wk}^n) \otimes \mathrm{id}_B)$. Hence, $f; (\mathtt{wk}^n \otimes \mathrm{id}_B); (\mathtt{eq}_n \otimes \mathrm{id}_B) = f$. For uniqueness, use the fact that $\mathtt{eq}_n$ is a left inverse for $\mathtt{wk}^n$. ◀

Thus, in any strong distributive sequoidally decomposable category, the diagram $\Delta(A)$ exists for any A. If a limit A^∞ for this diagram exists and is preserved by the tensor – i.e. for any B, $B \otimes A^\infty$ is the limit for $B \otimes \Delta(A)$ – then it is the cofree commutative comonoid.

Moreover, by distributivity, preservation by the tensor product implies preservation by the sequoid, which tells us that in this case the final sequence for $A \oslash \_$ must converge at ω and that the limit A^∞ must be the carrier for the final coalgebra for $A \oslash \_$.

3.2 Win-games and winning strategies

The construction from [23] covers a lot of important cases, but there are some situations in which the right conditions are not satisfied. Instead, we shall need the techniques that we shall describe in the following sections. One example in which we cannot use the Melliès-Tabareau-Tasson formula is that of *win-games*, or games with a winning condition [1, 8]. Given a game A, we write $\overline{P_A}$ to be the limit-closure of P_A – that is, P_A together with the set of infinite sequences, all of whose finite prefixes are in P_A. A *win-game* is a game A together with a function $\zeta_A \colon \overline{P_A} \to \{O, P\}$ such that:

- $\zeta_A(\epsilon) = P$
- $\zeta_A(sa) = \lambda_A(a)$

Thus, ζ_A is entirely determined on P_A, and the only new information is the values that ζ_A takes on the infinite positions in $\overline{P_A}$. The reason we bother to define ζ_A on finite positions at all is so that we can define it easily on the connectives:

$$\zeta_{A \otimes B}(s) = \zeta_A(s|_A) \wedge \zeta_B(s|_B) \qquad \zeta_{A \multimap B}(s) = \zeta_A(s|_A) \Rightarrow \zeta_B(s|_B)$$

$$\zeta_{\prod_{i \in I} A_i}(s) = \textstyle\bigwedge_{i \in I} \zeta_{A_i}(s|_{A_i}) \quad \zeta_{A \oslash B}(s) = \zeta_A(s|_A) \wedge \zeta_B(s|_B) \quad \zeta_{!A}(s) = \textstyle\bigwedge_{i \in I}(\zeta_A(s|_i))$$

Here, $\wedge$ and $\Rightarrow$ are the usual propositional connectives on $\{T, F\}$, where we identify T with P and F with O. The infinite positions s with $\zeta_A(s) = P$ are the *P-winning* positions, while the infinite positions s with $\zeta_A(s) = O$ are the *O-winning* positions.

We define a *winning strategy* on (A, ζ_A) to be a total strategy σ on A such that every infinite sequence arising as the limit of sequences in σ is a P-winning position. It is known

(see [1]) that the composition of winning strategies is winning and that we get a decomposable, distributive sequoidal category $\mathcal{W}$ with $!A$ as the final coalgebra for $A \oslash \_$ and the cofree commutative comonoid over A [8].

However, in this case, $!A$ is not the sequential limit of the symmetrized tensor powers over A. Since $\mathcal{W}$ is a decomposable, strong distributive sequoidal category, the symmetrized tensor powers of A are given by the sequoidal powers $A^{\oslash n}$. But now the limit of these objects is not quite the game $!A$; instead, it is the game $A^{\oslash\omega} = ¡A$ in which player O may open an arbitrarily large finite number of copies of A, but loses if he opens infinitely many. In the finite case, there was no way to keep track of infinite positions, so we could not make this distinction, but in the win-games case we can: we set $\zeta_{¡A}(s) = \zeta_{!A}(s)$, unless s contains moves in infinitely many games, in which case we set $\zeta_{¡A}(s) = P$.

This limit is not preserved by the functor $A \oslash \_$: in the game $A^{\oslash(\omega+1)} = A \oslash ¡A$, player O wins if he wins either in A or in $¡A$, so he can win even if he plays in infinitely many games, as long as he wins in the first copy of A. Similarly, in the game $A^{\oslash(\omega+n)}$, player O wins as long as he wins in one of the first n copies of A or opens finitely many copies. Therefore, the limit $A^{\oslash\omega 2}$ *is* the game $!A$: the final sequence for $A \oslash \_$ in $\mathcal{W}$ stabilizes at $\omega 2$ and, consequently, the exponential in $\mathcal{W}$ is not an MTT-exponential.

This example is a special case of our later result on transfinite games. For now, we shall examine a coalgebraic approach that will prove that the final coalgebra $!A$ for $A \oslash \_$ in the category $\mathcal{W}$ of win-games gives us a cofree commutative comonoid.

3.3 The coalgebraic construction under the strong monoidal hypothesis

We shall now need to assume that we are in a decomposable, distributive sequoidal category $(\mathcal{C}, \mathcal{C}_s, J, \mathtt{wk})$ such that $\mathcal{C}_s$ has all products and J preserves them. However, we shall no longer need the MTT assumption that the exponential should be constructed as a limit of sequoidal powers. The main cost is that we shall need to make a further assumption: that a certain naturally defined morphism $!A \otimes !B \to !(A \times B)$ is an isomorphism. This assumption, broadly corresponding to the demand that the functor $!A$ be strong monoidal from the Cartesian category $(\mathcal{C}, \times, 1)$ to the monoidal category $(\mathcal{C}, \otimes, I)$, will allow us to construct the comultiplication directly from the Cartesian structure and the definition of $!A$ as a final coalgebra.

▶ **Notation 7.** We shall sometimes make the monoidal structure of the Cartesian product explicit by writing $\sigma \times \tau$ for $\langle \mathrm{pr}_1; \sigma, \mathrm{pr}_2; \tau \rangle$.

▶ **Definition 8.** Let A, B be objects of an decomposable, distributive sequoidal category $(\mathcal{C}, \mathcal{C}_s, J, \mathtt{wk})$ with final coalgebras $!A \xrightarrow{\alpha_A} A \oslash !A$ for all endofunctors of the form $A \oslash \_$. Let A, B be objects of C. Then we have a composite $\kappa_{A,B}$:

$$\begin{aligned}
\kappa_{A,B} = !A \otimes !B &\xrightarrow{\qquad\qquad\mathtt{dec}_{A,B}\qquad\qquad} (!A \oslash !B) \times (!B \oslash !A) \\
\cdots &\xrightarrow{(\alpha_A \oslash \mathrm{id}_{!B}) \times (\alpha_B \oslash \mathrm{id}_{!A})} ((A \oslash !A) \oslash !B) \times ((B \oslash !B) \oslash !A) \\
\cdots &\xrightarrow{\mathtt{passoc}^{-1}_{A,!A,!B} \times \mathtt{passoc}^{-1}_{B,!B,!A}} (A \oslash (!A \otimes !B)) \times (B \oslash (!B \otimes !A)) \\
\cdots &\xrightarrow{\mathrm{id}_{A \oslash (!A \otimes !B)} \times (\mathrm{id}_B \oslash \mathtt{sym}_{!B,!A})} (A \oslash (!A \otimes !B)) \times (B \oslash (!A \otimes !B))
\end{aligned}$$

We get a morphism $\kappa_{A,B}; \mathtt{dist}^{-1} \colon !A \otimes !B \to (A \times B) \oslash (!A \otimes !B)$ and we write $\mathtt{coh}_{A,B} = (\!| \kappa_{A,B}; \mathtt{dist}^{-1} |\!) \colon !A \otimes !B \to !(A \times B)$.

▶ **Proposition 9.** *In the category of games, the morphism* $\mathtt{coh}_{A,B}$ *is an isomorphism for all games* A, B.

Proof. Observe that the morphism $\mathtt{coh}_{A,B}$ is the copycat strategy on $!A \otimes !B \multimap !(A \times B)$ that starts a copy of A on the left whenever a copy of A is started on the right and starts a copy of B on the left whenever a copy of B is started on the right (indeed, the morphisms in the diagram above are all copycat morphisms, so the copycat strategy we have just described must make that diagram commute). Since there are infinitely many copies of both A and B available in $!(A \times B)$, and since a new copy of A or B may be started at any time, we may define an inverse copycat strategy on $!(A \times B) \multimap !A \otimes !B$. ◀

Our first main result for this section will be the following:

▶ **Theorem 10.** *Let $(\mathcal{C}, \mathcal{C}_s, J, \mathtt{wk})$ be a distributive and decomposable sequoidal category with a final coalgebra $!A \xrightarrow{\alpha_A} A \oslash !A$ for each endofunctor of the form $A \oslash \_$. Suppose further that the morphism $\mathtt{coh}_{A,B}$ as defined above is an isomorphism for all objects A, B. Then $A \mapsto !A$ gives rise to a strong symmetric monoidal functor from the monoidal category $(\mathcal{C}, \times, 1)$ to the monoidal category $(\mathcal{C}, \otimes, I)$.*

We start off by defining a morphism $\mu \colon !A \to !A \otimes !A$. This will turn out to be the comultiplication for the cofree commutative comonoid over A. First, we note that we have the following composite:

$$!A \xrightarrow{\alpha_A} A \oslash !A \xrightarrow{\Delta} (A \oslash !A) \times (A \oslash !A) \xrightarrow{\mathtt{dist}^{-1}} (A \times A) \oslash !A$$

where Δ is the diagonal map on the product. We set $\sigma_A = \mathfrak{C}\, \alpha_A; \Delta; \mathtt{dist}^{-1} \,\mathfrak{D} \colon !A \to !(A \times A)$ and we set $\mu_A = \sigma_A; \mathtt{coh}^{-1}_{A,A} \colon !A \to !A \otimes !A$.

We also define a morphism $\mathtt{der}_A \colon !A \to A$. Note that since I is isomorphic to 1, we have a unique morphism $*_A \colon A \to I$ for each A. We define $\mathtt{der}_A$ to be the composite

$$!A \xrightarrow{\alpha_A} A \oslash !A \xrightarrow{\mathrm{id}_A \oslash *_{!A}} A \oslash I \xrightarrow{\mathtt{r}_A} A$$

We define the action of ! on morphisms as follows: suppose that $f \colon A \to B$ is a morphism in $\mathcal{C}$. Then we have a composite

$$!A \xrightarrow{\mu_A} !A \otimes !A \xrightarrow{\mathtt{der}_A \otimes \mathrm{id}_{!A}} A \otimes !A \xrightarrow{f \otimes \mathrm{id}_{!A}} B \otimes !A \xrightarrow{\mathtt{wk}_{B,!A}} B \oslash !A$$

and so we may define $!f$ to be the anamorphism $\mathfrak{C}\, \mu_A; \mathtt{der}_A \otimes \mathrm{id}_{!A}; f \otimes \mathrm{id}_{!A}; \mathtt{wk}_{B,!A} \,\mathfrak{D} \colon !A \to !B$.

▶ **Proposition 11.** *$f \mapsto !f$ respects composition, so ! is a functor. Moreover, ! is a strong symmetric monoidal functor from the Cartesian category $(\mathcal{C}, \times, 1)$ to the symmetric monoidal category $(\mathcal{C}, \otimes, I)$, witnessed by $\mathtt{coh}$ and ϵ, where ϵ is the anamorphism of the composite $I \xrightarrow{runit} I \otimes I \xrightarrow{*\otimes \mathrm{id}} 1 \otimes I \xrightarrow{\mathtt{wk}} 1 \oslash I$ (this composite is an isomorphism, so ϵ is as well).*

Proof. See Appendix to full version [10]. ◀

This completes the proof of Theorem 10.

Since ! is a strong monoidal functor, it induces a functor CCom(!) from the category $\mathrm{CCom}(\mathcal{C}, \times, 1)$ of comonoids over $(\mathcal{C}, \times, 1)$ to the category $\mathrm{CCom}(\mathcal{C}, \otimes, I)$ of comonoids over $(\mathcal{C}, \otimes, I)$ making the following diagram commute:

$$\begin{array}{ccc} \mathrm{CCom}(\mathcal{C}, \times, 1) & \xrightarrow{\mathcal{F}} & (\mathcal{C}, \times, 1) \\ {\scriptstyle \mathrm{CCom}(!)}\downarrow & & \downarrow{\scriptstyle !} \\ \mathrm{CCom}(\mathcal{C}, \otimes, I) & \xrightarrow{\mathcal{F}} & (\mathcal{C}, \otimes, I) \end{array}$$

where $\mathcal{F}$ is the forgetful functor.

Let A be an object of $\mathcal{C}$. Since $(\mathcal{C}, \times, 1)$ is Cartesian, the diagonal map $\Delta\colon A \to A \times A$ is the cofree commutative comonoid over A in $(\mathcal{C}, \times, 1)$.

▶ **Proposition 12.** $\mathrm{CCom}(!)\left(A \xrightarrow{\Delta} A \times A\right)$ *has comultiplication given by* $\mu_A\colon !A \to !A \otimes !A$ *and counit given by the unique morphism* $\eta_A\colon !A \to I$.

In particular, this proves that the comultiplication μ_A is associative and that the counit η_A is a valid counit for μ_A.

We can now state our second main result from this section.

▶ **Theorem 13.** *Let* $(\mathcal{C}, \mathcal{C}_s, J, \mathtt{wk})$ *be a sequoidal category satisfying all the conditions from Theorem 10. Let* A *be an object of* $\mathcal{C}$ *(equivalently, of* $\mathcal{C}_s$*). Then* $!A$*, together with the comultiplication* μ_A *and counit* η_A*, is the cofree commutative comonoid over* A*.*

3.4 The Sequoidal Exponential as a Bifree Algebra

Observe that in our category of games, $(!A, \alpha)$ is in fact a *bifree algebra* for $A \oslash \_$ – the isomorphism $\alpha^{-1} : A \oslash !A \to !A$ is an initial algebra for $A \oslash \_$. We may show that in such cases, the condition that ! is strong monoidal – and thus the cofree exponential – always holds[5] : we may define an inverse to $\mathtt{coh} : !A \otimes !B \to !(A \times B)$ as the *catamorphism* of the $A \oslash \_$-algebra:

$$(\kappa_{A,B}; \mathtt{dist})^{-1}\colon (A \times B) \oslash (!A \otimes !B) \to !A \otimes !B$$

It is not necessary for the final $A \oslash \_$-coalgebra to be bifree for the exponential to be strong monoidal and thus the cofree commutative comonoid. An example is provided by the category $\mathcal{W}$ of win-games and winning strategies, which is *sequoidal closed* (the restriction of the functor $A \multimap \_$ to strict strategies is right adjoint to $\_ \oslash A$, and inclusion sends this adjunction to the usual monoidal closure). To show that the final $A \oslash \_$-coalgebra in this category is not bifree, it suffices to observe that from such an algebra, we may derive a *fixed point* operator $\mathsf{fix}_A : \mathcal{C}(A, A) \to \mathcal{C}(I, A)$ for each A, such that $\mathsf{fix}_A(f); f = \mathsf{fix}_A(f)$.

▶ **Proposition 14.** *Suppose* $\mathcal{C}$ *is sequoidal closed and decomposable, and* $(!A, \alpha)$ *is a bifree* $A \oslash \_$*-algebra. Then we may define a fixed point operator on* A*.*

Proof. For any A, let $\Phi_A : !(A \multimap A) \to A$ be the catamorphism of the counit to the adjunction $A \oslash \_ \dashv A \multimap$, $\epsilon_{A,A} : (A \multimap A) \oslash A \to A$, which is a $(A \multimap A) \oslash \_$-algebra. For any morphism $f : A \to A$ we may define $\mathsf{fix}_A(f) = !\Lambda(f); \Phi_A$, where $\Lambda(f) : I \to (A \multimap A)$ is the "name" of f. ◀

As one would expect, it is not possible to define a fixed point operator on the category of games and winning strategies – for example, if $\bot$ is the game with a single move then the hom-set $\mathcal{C}(I, \bot)$ is empty and hence there can be no morphism $\mathsf{fix}_\bot(\mathrm{id}_\bot)$. So the final $A \oslash \_$-coalgebra is not bifree in this case.

---

[5] Without requiring our sequoidally decomposable category to have finite products we may equip each object $!A$ with the structure of a comonoid by defining: $\mu : !A \to !A \otimes !A$ to be the catamorphism of the $A \oslash \_$ algebra:

$$A \oslash (!A \otimes !A) \to A \oslash (!A \otimes !A) \times A \oslash (!A \otimes !A) \cong (A \oslash !A) \oslash !A \times (A \oslash !A) \oslash !A \cong (!A \oslash !A) \times (!A \oslash !A) \cong (!A \otimes !A)$$

This satisfies the further requirements of a *linear category* in the sense of [6], although it does not appear to be possible to show that it is the cofree commutative comonoid.

4 Transfinite Games

Of the conditions that we used to construct the cofree commutative comonoid in sequoidal categories, the requirement that $\mathtt{coh}_{A,B}$ be an isomorphism stands out as the least satisfactory. All the other conditions are 'finitary', and relate directly to the connectives we have introduced, whereas the morphism $\mathtt{coh}_{A,B}$ can only be constructed using the final coalgebra property for the exponential connective !. For this reason, we might wonder whether we can do without the condition that $\mathtt{coh}_{A,B}$ be an isomorphism. In this section, we shall give a negative answer to that question: we shall construct a distributive and decomposable sequoidal closed category with final coalgebras $!A$ for all functors of the form $A \oslash \_$, and shall show that $!A$ does not have a natural comonoid structure. In doing this, we hope to shed some light upon alternative algebraic or coalgebraic constructions for the cofree commutative comonoid that work in a purely 'finitary' manner.

▶ **ϟ Warning ϟ.** *Although we shall refer to the final coalgebra for $A \oslash \_$ as $!A$ to avoid introducing new notation, this object will not be the carrier for a linear exponential comonad (i.e., a model of the exponential from Linear Logic) in the category.*

Our sequoidal category will be closely modelled upon the category of games we have just considered: the objects will be games, with the modification that sequences of moves may now have transfinite length. This is a natural construction, occurring in the study of determinacy by Mycielski [24], Blass [7] and Weiss [12]. Transfinite games were used by Berardi and de'Liguoro to give a characterization of total functionals [5], and they appear to the authors to be present in the semantic context in the work of Roscoe [25], Levy [21] and Laird [18].

The general idea is as follows: we will show that the definition of the final coalgebra for the sequoid functor in a category of transfinite games is largely unchanged from the definition in the category of games with finite-length plays: $!A$ is the game formed from a countably infinite number of copies of A, indexed by ω, with the proviso that player O must open them in order. We observe that the copycat strategy $\mathtt{coh}_{A,B} : !A \otimes !B \to !(A \times B)$ is not an isomorphism, and that we cannot construct the comultiplication $!A \to !A \otimes !A$ in a sensible way. Moreover, we cannot construct the comonad $!A \to !!A$, so ! does not give us a model of linear logic in even the most general sense. In all three cases, the reason why the construction fails is that we might run out of copies of the game A (or B) on the left hand side before we have run out of copies on the right hand side. In the finite-plays setting, it is impossible to run out of copies of a subgame, because there are infinitely many copies, so it is impossible to play in all of them in a finite-length play. In the transfinite setting, however, we cannot guarantee this: consider, for example, a position in $!A_0 \multimap !A_1 \otimes !A_2$ (with indices given so we can refer to the different copies of A) in which player O has opened all the copies of A in $!A_1$. Since player P is playing by copycat, she must have opened all of the copies of A in $!A_0$. If, at time $\omega + 1$, player O now plays in $!A_2$, player P will have no reply to him.

The 'correct' definition of $!A$ in the transfinite game category is one in which there is an unlimited number of copies of A to open (rather than ω-many), but this is not the final coalgebra for the functor $A \oslash \_$.

4.1 Transfinite Games

We give a brief summary of the construction of the category of transfinite games.

We shall fix an additively indecomposable ordinal $\alpha = \omega^\beta$ throughout, which will be a bound on the ordinal length of positions in our game. So, for example, the original category

of games is the case $\alpha = \omega$. If X is a set, we write $X^{*<\alpha}$ for the set of transfinite sequences of elements of X of length less than α.

▶ **Definition 15.** A (completely negative) *game* or a *game over* α or an α*-game* is given by a forest (i.e., a prefix-closed set) P_A of alternating transfinite sequences of O-moves and P-moves of length less than α and a function $\zeta_A \colon P_A \to \{O, P\}$ that designates each position as an O-position or a P-position. We require that sa is a P-position if a is a P-move and an O-position if a is an O-move, so ζ_A only gives us information about plays of limiting length.

P_A is subject to a continuity condition: if s is a sequence of moves whose length is a limit ordinal and $t \in P_A$ for all proper prefixes $t \sqsubsetneq s$, then $s \in P_A$.

We say that a game A is *completely negative* if every position of limiting length is a P-position.

▶ **Definition 16.** A *strategy* for an α-game A is a non-empty prefix-closed subset of P_A that satisfies closure under O-replies and the determinism condition, just as for finite strategies.

We can form the product, tensor product, sequoid, exponential and linear implication in the same way that we do for finite games. The ζ-functions are extended to connectives according to the propositional formulae given for win-games above. If A and B are completely negative, then so are $A \times B$, $A \otimes B$, $A \oslash B$ and $!A$, but $A \multimap B$ might not be completely negative.

Given games A, B, C, strategies σ for $A \multimap B$ and τ for $B \multimap C$, we may compose σ and τ in the same way that we compose strategies for finite games (but we have to use the fact that α is additively indecomposable so that we can ensure that the interleaving of sequences of length less than α still has length less than α).

We can show that this composition is associative and moreover that we obtain a distributive and decomposable sequoidal category whose objects are completely negative games. We call this category $\mathcal{G}(\alpha)$ and call the corresponding strict subcategory $\mathcal{G}_s(\alpha)$. The hardest part of this is showing that the category is monoidal closed, because the linear implication of completely negative games is not necessarily completely negative. It turns out that there is always a 'minimal' completely negative game extending $A \multimap B$, which gives us the monoidal closed structure, but we will not discuss this here, since monoidal closedness is not particularly important to any of our constructions.

4.2 The final sequence for the sequoidal exponential

We now want to show that $\mathcal{G}(\alpha)$ has a final coalgebra for each functor $A \oslash \_$, given by the transfinite game $!A$, which is defined as follows:

- $M_{!A} = M_A \times \omega$
- $\lambda_{!A} = \lambda_A \circ \mathrm{pr}_1$

We define $!P_A$ to be the set of all sequences $s \in M_{!A}^{*<\alpha}$ such that $s|_n \in P_A$ for all n and such that every move in A_{n+1} occurs later than some move in A_n. Then we define $\zeta_{!A} \colon !P_A \to \{O, P\}$ by

$$\zeta_{!A}(s) = \bigwedge_{n \in \omega} \zeta_A(s|_n)$$

In other words, $\zeta_{!A}(s) = P$ if and only if $\zeta_A(s|_n) = P$ for all n. We define $P_{!A}$ to be the set of all sequences in $!P_A$ that are alternating with respect to $\zeta_{!A}$.

There is a natural copycat strategy $\alpha_A \colon !A \to A \oslash !A$, just as in the finite plays case. We want to show that this is the final coalgebra for $A \oslash \_$. The proof for the finite case found in

[8] will not work in this case, since it implicitly uses the fact that $!A$ is an MTT-exponential. In the transfinite categories, this is no longer the case.

While it is possible to prove that $\alpha_A \colon !A \to A \oslash !A$ is the final coalgebra for $A \oslash \_$ directly, we shall instead give a proof by extending the MTT sequence to the full final sequence. We shall give a complete classification of the games $A^{\oslash\gamma}$ and use it to show that the final sequence for $A \oslash \_$ must stabilize at $!A$.

▶ **Definition 17.** Let $s \in \omega^{*<\alpha}$ be any transfinite sequence of natural numbers. We define the *derivative* Δs of s to be the sequence given by removing all instances of 0 from s and subtracting 1 from all other terms. In other words, if $s \colon \gamma \to \omega$, for $\gamma < \alpha$, then we have:

$$\Delta s = s^{-1}(\omega \setminus \{0\}) \xrightarrow{s} \omega \setminus \{0\} \xrightarrow{-1} \omega$$

(where $s^{-1}(\omega \setminus \{0\})$ carries the induced order). We now define predicates $\_ \leq \gamma$ on sequences $s \in \omega^{*<\alpha}$ as follows:

- $\epsilon \leq 0$
- If $\Delta s \leq \gamma$, then $s \leq \gamma + 1$
- If μ is a limit ordinal and $s \in \omega^{*<\alpha}$ is such that for all successor-length prefixes $t \sqsubseteq s$ we have $t \leq \gamma$ for some $\gamma < \mu$, then $s \leq \mu$. In other words, $\{s \in \omega^{*<\alpha} : s \leq \mu\}$ is the limit-closure of the union of the sets $\{s \in \omega^{*<\alpha} : s \leq \gamma\}$ for $\gamma < \mu$.

It is easy to prove some basic results about these predicates:

▶ **Proposition 18.** *i) If $s \leq \gamma$ and t is any subsequence of s (not necessarily an initial prefix), then $t \leq \gamma$.*

ii) If $s \leq \gamma$, then $\Delta s \leq \gamma$

iii) If $s \leq \gamma$ and $\gamma \leq \delta$, then $s \leq \delta$

iv) If $s \in \omega^{*<\alpha}$ has length μ, where μ is a limit ordinal, then $s \leq \mu$. If s has length $\mu + n$ for some $n \in \omega$, then $s \leq \mu + \omega$. In particular, $s \leq \alpha$ for all $s \in \omega^{*<\alpha}$.

Proof. Left as an exercise. ◀

We can then classify the terms of the final sequence for $A \oslash \_$ as follows:

▶ **Theorem 19.** *Let A be any game. Then $A^{\oslash\gamma} \cong (M_{!A}, \lambda_{!A}, \zeta_{!A}, P_{!A,\gamma})$, where*

$$P_{!A,\gamma} = \{s \in P_{!A} : \ \mathrm{pr}_2 \circ s \leq \gamma\}$$

The morphism j^{δ}_{γ} is the copycat strategy.

▶ **Corollary 20.** *The final sequence for $A \oslash \_$ stabilizes at α and we have $A^{\oslash\alpha} = !A$.*

Proof. By Proposition 18(iv), $\mathrm{pr}_2 \circ s \leq \alpha$ for all $s \in P_{!A}$ and so $\mathrm{pr}_2 \circ s \leq (\alpha + 1)$, by Proposition 18(iii). It follows, by Theorem 19, that $A^{\oslash\alpha} = !A$ and that the morphism $A^{\oslash\alpha} \to A^{\oslash(\alpha+1)}$ is the morphism α_A. ◀

In particular, $!A$ is the carrier of the final coalgebra for $A \oslash \_$. But, as we saw before, it is not the carrier for the cofree commutative comonoid over A; indeed, more is true:

▶ **Proposition 21.** *$!A$ does not carry the structure of a linear exponential comonad.*

Proof. Indeed, if it did, then [26] we would have an isomorphism

$$!(A \times B) \cong !A \otimes !B$$

But this isomorphism does not hold for suitably long transfinite games. For example, if A and B are bounded games (i.e., games such that the lengths of plays are bounded by some finite number n) containing at least one play of length 2 then $!A \otimes !B$ contains plays of length $\omega 2$ (play through all the copies of A, then through all the copies of B), while the lengths of plays in $!(A \times B)$ are bounded by ω, since $A \times B$ is a bounded game. ◀

References

1 Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *The Journal of Symbolic Logic*, 59(2):543–574, 1994. URL: `http://arxiv.org/abs/1311.6057`.

2 Samson Abramsky and Guy McCusker. Full abstraction for Idealized Algol with passive expressions. *Theor. Comput. Sci.*, 227(1-2):3–42, September 1999. `doi:10.1016/S0304-3975(99)00047-X`.

3 S. Abramsky and G. McCusker. Call-by-value games. In M. Neilsen and W. Thomas, editors, *Proceedings of CSL '97*, pages 1–17. Springer-Verlag, 1998. `doi:10.1007/BFb0028004`.

4 S. Abramsky, K. Honda and G. McCusker. A fully abstract games semantics for general references. In *Proceedings of LICS '98*. IEEE Press, 1998. `doi:10.1109/LICS.1998.705669`.

5 Stefano Berardi and Ugo de'Liguoro. Total functionals and well-founded strategies. In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications: 4th International Conference, TLCA'99 L'Aquila, Italy, April 7–9, 1999 Proceedings*, pages 54–68. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999. `doi:10.1007/3-540-48959-2_6`.

6 G. M. Bierman. *What is a categorical model of Intuitionistic Linear Logic?*, pages 78–93. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995. `doi:10.1007/BFb0014046`.

7 Andreas Blass. Equivalence of two strong forms of determinacy. *Proceedings of the American Mathematical Society*, 52(1):373–376, 1975. URL: `http://www.jstor.org/stable/2040166`.

8 Martin Churchill, Jim Laird, and Guy McCusker. Imperative programs as proofs via game semantics. *CoRR*, abs/1307.2004, 2013. URL: `http://arxiv.org/abs/1307.2004`.

9 Pierre Clairambault. A remark on jim laird's games category for general references, 2008.

10 William John Gowers and James Laird. Sequoidal categories and transfinite games: A coalgebraic approach to stateful objects in game semantics. *CoRR*, abs/1706.00035, 2017. URL: `http://arxiv.org/abs/1706.00035`.

11 Martin Hyland. Game semantics. *Semantics and logics of computation*, 14:131, 1997.

12 Neeman Itay. *The Determinacy of Long Games*. De Gruyter CY, 2008. URL: `http://www.degruyter.com/view/product/178683`.

13 Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016.

14 G. Janelidze and G.M. Kelly. A note on actions of a monoidal category. *Theory and Applications of Categories [electronic only]*, 9:61–91, 2001. URL: `http://eudml.org/doc/122510`.

15 Yves Lafont. *Logiques, catégories et machines*. PhD thesis, Université Paris 7, 1988.

16 J. Laird. A categorical semantics of higher-order store. In *Proceedings of CTCS '02*, number 69 in ENTCS. Elsevier, 2002.

17 J. Laird. Locally boolean domains. *Theoretical Computer Science*, 342(1):132 – 148, 2005. `doi:10.1016/j.tcs.2005.06.007`.

18 J. Laird. Sequential algorithms for unbounded nondeterminism. *Electronic Notes in Theoretical Computer Science*, 319:271 – 287, 2015. doi:10.1016/j.entcs.2015.12.017.

19 James Laird. Functional programs as coroutines: A semantic analysis. *Logical methods in computer science*. To appear.

20 Joachim Lambek. A fixpoint theorem for complete categories. *Mathematische Zeitschrift*, 103(2):151–161, 1968. doi:10.1007/BF01110627.

21 Paul Blain Levy. Infinite trace equivalence. *Annals of Pure and Applied Logic*, 151(2):170 – 198, 2008. doi:10.1016/j.apal.2007.10.007.

22 G. McCusker. A fully abstract relational model of Syntactic Control of Interference. In *Proceedings of Computer Science Logic '02*, number 2471 in LNCS. Springer, 2002.

23 Paul-André Melliès, Nicolas Tabareau, and Christine Tasson. *An Explicit Formula for the Free Exponential Modality of Linear Logic*, pages 247–260. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-02930-1_21.

24 Jan Mycielski. On the axiom of determinateness. *Fundamenta Mathematicae*, 53(2):205–224, 1964. URL: http://eudml.org/doc/213736.

25 A. W. Roscoe. Unbounded non-determinism in CSP. *Journal of Logic and Computation*, 3(2):131, 1993. doi:10.1093/logcom/3.2.131.

26 Andrea Schalk. What is a categorical model for linear logic?, October 2004. URL: http://www.cs.man.ac.uk/~schalk/notes/llmodel.pdf.

27 James Worrell. On the final sequence of a finitary set functor. *Theoretical Computer Science*, 338(1):184 – 199, 2005. doi:10.1016/j.tcs.2004.12.009.

Free Constructions and Coproducts of d-Frames*†

Tomáš Jakl[1] **and Achim Jung**[2]

1 **Department of Applied Mathematics, Charles University, Prague, Czechia**
School of Computer Science, University of Birmingham, Birmingham, UK
jaklt@kam.mff.cuni.cz

2 **School of Computer Science, University of Birmingham, Birmingham, UK**
A.Jung@cs.bham.ac.uk

Abstract

A general theory of presentations for d-frames does not yet exist. We review the difficulties and give sufficient conditions for when they can be overcome. As an application we prove that the category of d-frames is closed under coproducts.

1998 ACM Subject Classification F.3.1 Logics of programs, F.3.2 Semantics of Programming Languages

Keywords and phrases Free construction, d-frame, coproduct, C-ideals

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.14

1 Introduction

In his celebrated *Domain Theory in Logical Form* [1], Abramsky describes a flexible framework for connecting the denotational semantics of a programming language with an algebraic presentation of a program logics. The denotational spaces are spectral spaces and the algebras are distributive lattices; they are connected via Stone duality [7].

The attempt to expand the scope of Abramsky's work to cover probabilistic and real-number computation led to the study of stably compact spaces and their Stone duality. Later, it was shown that stably compact spaces have a very natural bitopological description; they are exactly the compact regular bitopological spaces (or *bispaces* for short) [8, 11]. Moreover, the Stone-type duality between bispaces and *d-frames* given in [8] has a finitistic description in the compact regular case, and so we can try to extend Abramsky's work to this setting.

Free constructions of distributive lattices are an essential tool of Domain Theory in Logical Form and therefore a general theory of free constructions of d-frames is highly desirable. Unfortunately, no such theory exists as of yet. The difficulty lies in the mixed algebraic-relational nature of d-frames and particularly in the axiom (con-tot). In the absence of a general theory one can look at special instances of the problem where the difficulties with (con-tot) can be controlled. This is our approach in this paper.

The carrier of a d-frame is two-sorted, consisting of two standard frames L_+ and L_-. We use the usual generator and relations machinery to present them separately. The remaining parts of the structure, the consistency and totality relations $\mathsf{con}, \mathsf{tot} \subseteq L_+ \times L_-$ can be specified by generating relations con_1 and tot_1, but it is not clear how to make sure that (con-tot), the only axiom that bonds both relations, will hold in the generated structure.

* A full version of this paper can be found at https://arxiv.org/abs/1704.04029.

† The first author was supported by the grant SVV–2017–260452.

Rather than solve this general problem, we provide sufficient conditions which can be checked in an early stage of the generating process (Section 4).

As an application, we prove that the category of d-frames is closed under coproducts (Section 5). In forthcoming work, [5], we show that the same techniques allow us to define the d-frame that corresponds to the *Vietoris power-space* over a bispace. Together, this lays the foundation for the extension of Abramsky's program as explained above as well as four-valued coalgebraic logic (inspired by [9, 10, 4]). We believe that our results are interesting from a model-theoretic perspective as well, as they provide an example of a free construction for a two-sorted algebraic-relational structure. Although not completely general, our techniques hold promise for extending many other frame-theoretic constructions to d-frames (for examples see [6, 12, 13]).

2 Preliminaries

Frames are algebraic structures which capture the order-theoretic properties of the lattice of open sets of a topological space. We say that a complete lattice $(L; \bigvee, \wedge, 0, 1)$ is a *frame* if it satisfies the following infinitary distributivity law

(Frm) $b \wedge (\bigvee_i a_i) = \bigvee_i (b \wedge a_i)$.

The counterparts to continuous maps are the *frame homomorphisms* which are maps distributing over *all* joins and *all finite* meets.

A topological space $(X; \tau)$ gives rise to a frame: the lattice of its open sets ordered by set inclusion $\Omega(X) = (\tau; \bigcup, \cap; \emptyset, X)$ is a frame. Also, any continuous map $f\colon X \to Y$ gives rise to a frame homomorphism $\Omega(f)\colon \Omega(Y) \to \Omega(X)$ as $U \in \tau^Y \mapsto f^{-1}[U] \in \tau^X$.

Following the example of frames we have d-frames as the algebraic counterparts to bitopological spaces (or *bispaces*, for short)[1]. Because bispaces have two topologies, we expect to have two frames, L_+ and L_-, as part of the structure of d-frames.

This alone has some consequences. We can recognise two orders in the product $L_+ \times L_-$; the first is the *information order* $\sqsubseteq$ where, for $\alpha = (\alpha_+, \alpha_-)$ and $\beta = (\beta_+, \beta_-) \in L_+ \times L_-$, $\alpha \sqsubseteq \beta$ iff $\alpha_+ \leq \beta_+$ and $\alpha_- \leq \beta_-$. The second is the *logical order* $\leq$ where $\alpha \leq \beta$ iff $\alpha_+ \leq \beta_+$ and $\alpha_- \geq \beta_-$. Both $(L_+ \times L_-; \sqsubseteq)$ and $(L_+ \times L_-; \leq)$ are bounded distributive lattices with meets and joins computed as follows

$$\alpha \vee \beta = (\alpha_+ \vee \beta_+, \alpha_- \wedge \beta_-), \qquad \alpha \sqcup \beta = (\alpha_+ \vee \beta_+, \alpha_- \vee \beta_-),$$
$$\alpha \wedge \beta = (\alpha_+ \wedge \beta_+, \alpha_- \vee \beta_-), \qquad \alpha \sqcap \beta = (\alpha_+ \wedge \beta_+, \alpha_- \wedge \beta_-).$$

The smallest and largest elements in the information order are $\bot = (0, 0)$ and $\top = (1, 1)$, and in the logical order $\mathit{ff} = (0, 1)$ and $\mathit{tt} = (1, 0)$, respectively.

With just two frames we would not be able to express many bitopological properties. One can require L_+ and L_- to be subframes of a bigger frame representing the join of the two topologies as proposed by Banaschewski [3]. Or, following the second author and Moshier [8], we can require two binary relations con and tot between the two frame components where $(a, b) \in \mathsf{con}$ corresponds to a being disjoint from b, and $(a, b) \in \mathsf{tot}$ if a and b cover the whole space. Our work takes the second approach.

[1] Bispaces are the structures $(X; \tau_+, \tau_-)$ where $(X; \tau_+)$ and $(X; \tau_-)$ are topological spaces. A map between two bispaces $f\colon X \to Y$ is *bicontinuous* if both $f_+\colon (X; \tau_+^X) \to (Y; \tau_+^Y)$ and $f_-\colon (X; \tau_-^X) \to (Y; \tau_-^Y)$ (which are acting on the underlying set X the same way as f does) are continuous.

Formally, then, a *d-frame* is a structure $\mathcal{L} = (L_+, L_-; \mathsf{con}, \mathsf{tot})$ such that L_+ and L_- are frames and the binary *consistency* $\mathsf{con} \subseteq L_+ \times L_-$ and *totality* $\mathsf{tot} \subseteq L_+ \times L_-$ relations satisfy the following axioms, for all $\alpha, \beta \in L_+ \times L_-$:

(con–↓) $\alpha \in \mathsf{con}$ and $\beta \sqsubseteq \alpha \implies \beta \in \mathsf{con}$,
(tot–↑) $\alpha \in \mathsf{tot}$ and $\beta \sqsupseteq \alpha \implies \beta \in \mathsf{tot}$,
(con,tot–*tt*, *ff*) $tt \in \mathsf{con}$ and $tt \in \mathsf{tot}$, $ff \in \mathsf{con}$ and $ff \in \mathsf{tot}$,
(con–∧, ∨) $\alpha, \beta \in \mathsf{con} \implies \alpha \vee \beta \in \mathsf{con}$ and $\alpha \wedge \beta \in \mathsf{con}$,
(tot–∧, ∨) $\alpha, \beta \in \mathsf{tot} \implies \alpha \vee \beta \in \mathsf{tot}$ and $\alpha \wedge \beta \in \mathsf{tot}$,
(con–$\bigsqcup^\uparrow$) $A \subseteq \mathsf{con}$ and A is $\sqsubseteq$-directed $\implies \bigsqcup^\uparrow A \in \mathsf{con}$,
(con–tot) $\alpha \in \mathsf{con}, \beta \in \mathsf{tot}$ and ($\alpha_+ = \beta_+$ or $\alpha_- = \beta_-$) $\implies \alpha \sqsubseteq \beta$.

Algebraically speaking, the 3rd–6th axioms say that $(\mathsf{con}; \wedge, \vee, tt, ff)$ and $(\mathsf{tot}; \wedge, \vee, tt, ff)$ are (bounded) distributive lattices and that $(\mathsf{con}; \sqsubseteq)$ is a DCPO. Directed suprema are computed pointwise, i.e. for a $\sqsubseteq$-directed $A \subseteq \mathsf{con}$, $\bigsqcup^\uparrow A = (\bigvee^\uparrow \{\alpha_+ : \alpha \in A\},\ \bigvee^\uparrow \{\alpha_- : \alpha \in A\})$.

A pair of frame homomorphisms $h = (h_+ \colon L_+ \to M_+, h_- \colon L_- \to M_-)$ is a *d-frame homomorphism* $h \colon \mathcal{L} \to \mathcal{M}$ if, for all $\alpha \in \mathsf{con}^{\mathcal{L}}$, $h(\alpha) = (h_+(\alpha_+), h_-(\alpha_-)) \in \mathsf{con}^{\mathcal{M}}$ and, for all $\alpha \in \mathsf{tot}^{\mathcal{L}}$, $h(\alpha) \in \mathsf{tot}^{\mathcal{M}}$.

Every bispace $X = (X; \tau_+, \tau_-)$ gives rise to a d-frame $\Omega^d(X) = (\tau_+, \tau_-; \mathsf{con}^X, \mathsf{tot}^X)$ where $(U_+, U_-) \in \mathsf{con}^X$ iff $U_+ \cap U_- = \emptyset$ and $(U_+, U_-) \in \mathsf{tot}^X$ iff $U_+ \cup U_- = X$. Similarly, every bicontinuous map $f \colon X \to Y$ gives rise to a d-frame homomorphism $\Omega^d(f) = (\Omega(f_+), \Omega(f_-)) \colon \Omega^d(Y) \to \Omega^d(X)$.

The (con-tot) axiom, while essential in the theory of d-frames, is harder to guarantee in constructions. We therefore introduce the auxiliary notion of a *pre-d-frame* where all but the (con-tot) axiom of d-frames are required to hold.

▶ Remark. Often, when we quantify over elements or sets that appear in both plus and minus forms we will use the symbol "±" to mean both of them. For example, "$A_\pm$ has property X" means "A_+ and A_- have property X", or "there exist elements $x_\pm \in L_\pm$" means "there exist elements $x_+ \in L_+$ and $x_- \in L_-$".

Also, because of the symmetrical nature of d-frames, many proofs consist of two identical arguments, one for the plus and and one for the minus side. Instead, we give only one of the variants without even mentioning the other.

3 Presentations

3.1 Presentation of frames

Frames, like other algebraic structures, may be presented in terms of generators and relations $\langle G|R\rangle$. The resulting frame $\mathbb{F}r\langle G|R\rangle$ is obtained as the quotient $\mathbb{F}r\langle G\rangle/_{\sim_R}$. Here, $\mathbb{F}r\langle G\rangle$ represents the term algebra generated by the set of generators G which, because of the frame distributivity law, consists of terms of the form: $\bigvee_i (\wedge_{j=1}^{n_i}\ g_{i,j})$. The congruence $\sim_R$ is generated from a relation $R \subseteq \mathbb{F}r\langle G\rangle \times \mathbb{F}r\langle G\rangle$ where each element of R is thought of as an equation:

$$\bigvee_i (\wedge_{j=1}^{n_i}\ g_{i,j}) = \bigvee_{i'} (\wedge_{j'=1}^{n'_{i'}}\ g'_{i',j'}). \tag{1}$$

However, the structure of $\mathbb{F}r\langle G\rangle/_{\sim_R}$ is not transparent at all. Its elements are equivalence classes of infinitary terms quotiented by R, which itself consists of "infinitary" equations. This is addressed in the $\mathcal{C}$-ideal presentation of frames. We assume that our generators form

a meet-semilattice[2] B representing the terms $\wedge_{j=1}^{n} g_j$. Moreover, we can restrict to equations in which the right-hand side consists of a single finite meet of generators, i.e. an element of B. In the terminology of $\mathcal{C}$-ideals, we have a set of cover relations $\mathcal{C}$ where a *cover relation* is any pair $U \dashv a$ such that $a \in B$ and $U \subseteq {\downarrow}a$ (to represent the equation $\bigvee U = a$). If, moreover, $\mathcal{C}$ satisfies the stability condition

$$U \dashv a \in \mathcal{C},\ b \leq a \implies \{u \wedge b : u \in U\} \dashv b \in \mathcal{C} \qquad (\mathcal{C}\text{-st.})$$

then we call $(B, \mathcal{C})$ a *frame presentation.*

The frame presented by $(B, \mathcal{C})$ has an explicit description as the frame of all $\mathcal{C}$-ideals, denoted by $\mathcal{C}\text{-}\mathtt{Idl}(B)$, where $I \subseteq B$ is a *$\mathcal{C}$-ideal* if it is a downset and

$$U \dashv a \in \mathcal{C},\ U \subseteq I \implies a \in I$$

Computing with $\mathcal{C}$-ideals is straightforward. The join of a set $\{I_i\}_i$ of $\mathcal{C}$-ideal is computed as $\mathcal{C}\text{-}\mathtt{Idl}\langle \bigcup_i I_i \rangle$ where, for an $M \subseteq B$, $\mathcal{C}\text{-}\mathtt{Idl}\langle M \rangle$ is the smallest $\mathcal{C}$-ideal containing M. The meets of $\mathcal{C}$-ideals are just intersections: $\bigwedge_i I_i = \bigcap_i I_i$, [6, Proposition II.2.11].

There is a map translating syntactic terms to their semantic interpretation as $\mathcal{C}$-ideals with the following universal property:

▶ **Lemma 1** (Universality). *Let $(B, \mathcal{C})$ be a presentation of a frame. Then the map $[\![-]\!] \colon B \to \mathcal{C}\text{-}\mathtt{Idl}(B)$ defined as $b \mapsto \mathcal{C}\text{-}\mathtt{Idl}\langle\{b\}\rangle$ is a meet-semilattice homomorphism that transforms covers into joins, i.e. $\bigvee\{[\![u]\!] : u \in U\} = [\![a]\!]$ for every $U \dashv a \in \mathcal{C}$.*

Moreover, $[\![-]\!]$ is universal among all such maps. That is, if $f \colon B \to L$ is a meet-semilattice homomorphisms that transform covers in $\mathcal{C}$ into joins, where L is a frame, then there exists a unique frame homomorphism $\overline{f} \colon \mathcal{C}\text{-}\mathtt{Idl}(B) \to L$ such that $f = \overline{f} \circ [\![-]\!]$.

▶ Remark. There are numerous ways of presenting frames, e.g. [13], [6], [2] or [12]. We picked this one because it suits us better later on for the coproduct of d-frames. For the actual definition of presentation of d-frames it should not really matter as long as we have a universality property similar to the one in Lemma 1.

3.2 Presentations of pre-d-frames

In this section we show that we can extend the classical theory to also present a (pre-)d-frame $(L_+, L_-; \mathsf{con}, \mathsf{tot})$. Let us assume that $L_\pm = \mathcal{C}_\pm\text{-}\mathtt{Idl}(B_\pm)$, for some frame presentations $(B_\pm, \mathcal{C}_\pm)$, as in the previous section. We also have the translations $[\![-]\!]_\pm \colon B_\pm \to L_\pm$ from syntax to semantics according to Lemma 1.

Any consistency relation con on $L_+ \times L_-$ can be specified via the generators: Let $\alpha \in \mathsf{con}$. Since the sets $[\![B_\pm]\!]_\pm = \{[\![b]\!]_\pm : b \in B_\pm\}$ generate the frames $L_\pm$,

$$\alpha = (\bigvee_{i \in I_+} b^i_+,\ \bigvee_{i \in I_-} b^i_-) \quad \text{for some } \{b^i_+\}_i \subseteq [\![B_+]\!]_+ \text{ and } \{b^i_-\}_i \subseteq [\![B_-]\!]_-$$

and, because con is downwards closed in the information order, con must contain all the pairs $(b^i_+,\ b^{i'}_-)$, for $(i, i') \in I_+ \times I_-$. Moreover, the converse is also true:

▶ **Lemma 2.** $(\bigvee_{i \in I_+} b^i_+,\ \bigvee_{i \in I_-} b^i_-) \in \mathsf{con}$ *iff* $(b^i_+,\ b^{i'}_-) \in \mathsf{con}$, *for all* $(i, i') \in I_+ \times I_-$

[2] We always assume that meet-semilattices are closed under *all* finite meets, i.e. they also contain the top element.

Proof. Only the right-to-left implication remains to be proved. Assume that $(b^i_+, b^{i'}_-) \in \mathsf{con}$, for all $(i, i') \in I_+ \times I_-$. Since con is $\wedge$-closed, for an $i \in I_+$ and a finite $F_- \subseteq^{\text{fin}} I_-$, $(b^i_+, \bigvee_{i \in F_-} b^i_-) \in \mathsf{con}$. Similarly, since con is $\vee$-closed, for finite $F_- \subseteq^{\text{fin}} I_-$ and $F_+ \subseteq^{\text{fin}} I_+$, $(\bigvee_{i \in F_+} b^i_+, \bigvee_{i \in F_-} b^i_-) \in \mathsf{con}$. Notice that the set $M = \{(\bigvee_{i \in F_+} b^i_+, \bigvee_{i \in F_-} b^i_-) : F_+ \subseteq^{\text{fin}} I_+ \text{ and } F_- \subseteq^{\text{fin}} I_-\}$ is directed and $\bigsqcup^{\uparrow} M = (\bigvee_{i \in I_+} b^i_+, \bigvee_{i' \in I_-} b^{i'}_-)$. Moreover, because $M \subseteq \mathsf{con}$ and con is closed under directed suprema, $\bigsqcup^{\uparrow} M \in \mathsf{con}$. ◀

This means that we can specify con by a subset $\mathsf{con}_1 \subseteq B_+ \times B_-$ such that $\mathsf{con} = \mathtt{CON}\langle [\![\mathsf{con}_1]\!] \rangle$ where $\mathtt{CON}\langle [\![\mathsf{con}_1]\!] \rangle$ is *the smallest consistency relation* containing $[\![\mathsf{con}_1]\!] = \{([\![\alpha_+]\!]_+, [\![\alpha_-]\!]_-) : \alpha \in \mathsf{con}_1\}$.[3]

In general, we cannot hope to do the same for tot, i.e. find a $\mathsf{tot}_1 \subseteq B_+ \times B_-$ such that $\mathsf{tot} = \mathtt{TOT}\langle [\![\mathsf{tot}_1]\!] \rangle$ where $\mathtt{TOT}\langle [\![\mathsf{tot}_1]\!] \rangle$ is *the smallest totality relation* containing $[\![\mathsf{tot}_1]\!]$. We would have to specify tot by a subset of $\mathcal{P}(B_+) \times \mathcal{P}(B_-)$. However, the special kind of presentations, when $\mathsf{tot}_1 \subseteq B_+ \times B_-$, turns out to be sufficient for our purposes.

▶ **Definition 3.** A tuple $(B_+, B_-; \mathcal{C}_+, \mathcal{C}_-; \mathsf{con}_1, \mathsf{tot}_1)$ is a *presentation of a pre-d-frame* if
(d-Pres-1) $(B_+, \mathcal{C}_+)$ and $(B_-, \mathcal{C}_-)$ are presentations of frames,
(d-Pres-2) $\mathsf{con}_1 \subseteq B_+ \times B_-$ and $\mathsf{tot}_1 \subseteq B_+ \times B_-$.
The resulting pre-d-frame is obtained in two steps. First, we generate the frames of $\mathcal{C}$-ideals $\mathcal{C}_\pm\text{-}\mathtt{Idl}(B_\pm)$ and then we generate the consistency and totality relations from the embedded relations $[\![\mathsf{con}_1]\!], [\![\mathsf{tot}_1]\!] \subseteq \mathcal{C}_+\text{-}\mathtt{Idl}(B_+) \times \mathcal{C}_-\text{-}\mathtt{Idl}(B_-)$. We obtain the following pre-d-frame:

$$(\mathcal{C}_+\text{-}\mathtt{Idl}(B_+), \mathcal{C}_-\text{-}\mathtt{Idl}(B_-);\ \mathtt{CON}\langle [\![\mathsf{con}_1]\!] \rangle, \mathtt{TOT}\langle [\![\mathsf{tot}_1]\!] \rangle) \tag{gen.}$$

Similarly to its frame counterpart, $[\![-]\!] = ([\![-]\!]_+, [\![-]\!]_-)$ has the following universal property.

▶ **Lemma 4** (Universality). *Let $(B_+, B_-; \mathcal{C}_+, \mathcal{C}_-; \mathsf{con}_1, \mathsf{tot}_1)$ be a presentation of a pre-d-frame. Then,*

$$[\![-]\!] \colon (B_+, B_-; \mathsf{con}_1, \mathsf{tot}_1) \to (\mathcal{C}_+\text{-}\mathtt{Idl}(B_+), \mathcal{C}_-\text{-}\mathtt{Idl}(B_-);\ \mathtt{CON}\langle [\![\mathsf{con}_1]\!] \rangle, \mathtt{TOT}\langle [\![\mathsf{tot}_1]\!] \rangle),$$

is presentation preserving, i.e. its components are meet-semilattice homomorphisms that transform covers from $\mathcal{C}_\pm$ into joins and together they preserve con_1 and tot_1.

Also, if $\mathcal{M}$ is a pre-d-frame and $f = (f_+, f_-) \colon (B_+, B_-; \mathsf{con}_1, \mathsf{tot}_1) \to \mathcal{M}$ is a presentation-preserving pair of maps, then there is a unique d-frame homomorphism

$$\overline{f} \colon (\mathcal{C}_+\text{-}\mathtt{Idl}(B_+), \mathcal{C}_-\text{-}\mathtt{Idl}(B_-);\ \mathtt{CON}\langle [\![\mathsf{con}_1]\!] \rangle, \mathtt{TOT}\langle [\![\mathsf{tot}_1]\!] \rangle) \to \mathcal{M}$$

such that $f = \overline{f} \circ [\![-]\!]$. Moreover, the components of $\overline{f}$ are the unique frame homomorphisms that are guaranteed to exist by Lemma 1.

4 Generating d-frames

So far we made no attempt in making sure that the axiom (con-tot) is satisfied in the generated pre-d-frame. Let us fix a presentation $(B_+, B_-; \mathcal{C}_+, \mathcal{C}_-; \mathsf{con}_1, \mathsf{tot}_1)$ for the rest of this section and, because both frame components stay intact after we generate them, let us

[3] Formally, for an $R \subseteq L_+ \times L_-$,

$$\mathtt{CON}\langle R \rangle = \bigcap \{R' \subseteq L_+ \times L_- \mid R \subseteq R', R' \text{ is } {\downarrow}\text{-closed, closed under } \wedge, \vee, \textstyle\bigsqcup^{\uparrow}, \text{ and } \mathit{ff}, \mathit{tt} \in R'\}.$$

denote them by $L_\pm \stackrel{\text{def}}{\equiv} \mathcal{C}_\pm\text{-}\mathtt{Idl}(B_\pm)$. Also, for brevity, we will identify $B_\pm$ with $[\![B_\pm]\!]_\pm \subseteq L_\pm$ and, also, con_1 and tot_1 with $[\![\mathsf{con}_1]\!]$ and $[\![\mathsf{tot}_1]\!] \subseteq L_+ \times L_-$, respectively.

The question for this section is: Under which conditions for $(B_+, B_-; \mathcal{C}_+, \mathcal{C}_-; \mathsf{con}_1, \mathsf{tot}_1)$ is the generated pre-d-frame

$$(L_+, L_-; \mathtt{CON}\langle\mathsf{con}_1\rangle, \mathtt{TOT}\langle\mathsf{tot}_1\rangle)$$

a d-frame? We solve this problem (partially) by showing that the following conditions are sufficient (though not necessarily minimal):

1. ($\downarrow\mathsf{con}_{\wedge,\vee}$-ind$_\pm$), from Section 4.2, which will ensure that the structure of $\mathtt{CON}\langle\mathsf{con}_1\rangle$ is "sufficiently simple", and
2. ($\lambda^4_\pm$-con-tot), from Section 4.3, which is just a simple instance of (con-tot).

4.1 The structure of $\mathtt{CON}\langle\mathsf{con}_1\rangle$ and $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$

Before we get to the two conditions, we show that the relations $\mathtt{CON}\langle\mathsf{con}_1\rangle$ and $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$ can be generated more explicitly. As in the HSP theorem from universal algebra, we can close con_1 and tot_1 under the operations they should be closed under (e.g. $\wedge$, $\vee$, etc.) and, if we proceed in a certain order, we do not have to repeat any of the steps.

Let $R \subseteq L_+ \times L_-$ be a any relation. We say that R is $\wedge$-closed (resp. $\vee$-closed), if for every $\alpha, \beta \in R$, $\alpha \wedge \beta \in R$ (resp. $\alpha \vee \beta \in R$). By $\downarrow R$ denote the downwards closure of R in the $\sqsubseteq$-ordering, i.e. the relation $\{\alpha \in L_+ \times L_- \mid \exists \beta \in R.\ \alpha \sqsubseteq \beta\}$ and define $\uparrow R$ similarly.

Finally, define $\mathcal{D}(R) \stackrel{\text{def}}{\equiv} \{\bigsqcup^\uparrow A \mid A \subseteq^\uparrow R\}$.[4] Note that $\mathcal{D}(R)$ is only a *"one-step"* closure under joins of directed subsets in $\sqsubseteq$-order. $\mathcal{D}(R)$ might still contain directed subsets which do not have suprema in $\mathcal{D}(R)$. To close R under all directed suprema, one would have to iterate this process. However, as we will see later, there are natural conditions under which only one application is enough.

▶ **Lemma 5.** *Let L_+, L_- be two frames and let $R \subseteq L_+ \times L_-$ be a relation. Then:*

1. *If R is $(\wedge, \vee)$-closed then $\downarrow R$ and $\uparrow R$ in $L_+ \times L_-$ are also $(\wedge, \vee)$-closed.*
2. *If R is $(\wedge, \vee)$-closed then the relation $\mathcal{D}(R)$ is still $(\wedge, \vee)$-closed.*
3. *If R is downwards closed then the relation $\mathcal{D}(R)$ is still downwards closed.*

Proof. For 1., let $\alpha, \beta \in \downarrow R$. This means that there are $\alpha', \beta' \in R$ such that $\alpha \sqsubseteq \alpha'$ and $\beta \sqsubseteq \beta'$. Observe that $(\alpha \wedge \beta)_+ = \alpha_+ \wedge \beta_+ \leq \alpha'_+ \wedge \beta'_+ = (\alpha' \wedge \beta')_+$ and similarly $(\alpha \wedge \beta)_- \leq (\alpha' \wedge \beta')_-$. Therefore, $\alpha \wedge \beta \sqsubseteq \alpha' \wedge \beta' \in R$ and $\alpha \wedge \beta \in \downarrow R$. Proving closedness $\downarrow R$ under $\vee$ is the same and the same reasoning also applies to $\uparrow R$. For 2., let $\alpha, \beta \in \mathcal{D}(R)$. From the definition $\alpha = \bigsqcup^\uparrow_i \alpha^i$ and $\beta = \bigsqcup^\uparrow_j \beta^j$ for some α^i's and β^j's from R. Let us calculate,

$$\begin{aligned}
\alpha \wedge \beta &= (\textstyle\bigvee^\uparrow_i \alpha^i_+ \wedge \bigvee^\uparrow_j \beta^j_+, \bigvee^\uparrow_i \alpha^i_- \vee \bigvee^\uparrow_j \beta^j_-) \\
&= (\textstyle\bigvee^\uparrow_i \bigvee^\uparrow_j (\alpha^i_+ \wedge \beta^j_+), \bigvee^\uparrow_i \bigvee^\uparrow_j (\alpha^i_- \vee \beta^j_-)) \\
&= (\textstyle\bigvee^\uparrow_{i,j} (\alpha^i_+ \wedge \beta^j_+), \bigvee^\uparrow_{i,j} (\alpha^i_- \vee \beta^j_-))
\end{aligned}$$

Notice that the set $\{\alpha^i \wedge \beta^j : i \in I, j \in J\}$ is directed since $\{\alpha^i\}_i$ and $\{\beta^j\}_j$ are and, moreover, $\alpha^i \wedge \beta^j \in R$ for all i, j since R is closed under logical meets.

For 3., let $\beta \sqsubseteq \bigsqcup^\uparrow_i \alpha_i$ where α_i's are from R. Then, $\beta = \bigsqcup^\uparrow_i (\beta \sqcap \alpha_i) \in \mathcal{D}(R)$ because the set $\{\beta \sqcap \alpha_i\}_i$ is a directed subset of R. ◀

[4] $A \subseteq^\uparrow R$ means that A is a directed subset of R in the $\sqsubseteq$-order.

Lemma 5 shows the order in which one can generate $\mathtt{CON}\langle\mathsf{con}_1\rangle$ and $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$. Set $\mathsf{con}_{\wedge,\vee}$ to be the algebraic closure of con_1 under all *finite* logical joins and meets in $L_+ \times L_-$, and define $\mathsf{tot}_{\wedge,\vee}$ correspondingly. Then we have:

▶ **Corollary 6.**

$$\mathtt{CON}\langle\mathsf{con}_1\rangle = \bigcup_{\iota \in Ord} \mathcal{D}^\iota(\downarrow \mathsf{con}_{\wedge,\vee}) \quad \textit{and} \quad \mathtt{TOT}\langle\mathsf{tot}_1\rangle = \uparrow \mathsf{tot}_{\wedge,\vee}$$

where, for an ordinal ι and a limit ordinal λ,

$$D^0(R) \stackrel{def}{\equiv} R, \quad D^{\iota+1}(R) \stackrel{def}{\equiv} \mathcal{D}(\mathcal{D}^\iota(R)) \quad \textit{and} \quad \mathcal{D}^\lambda(R) \stackrel{def}{\equiv} \bigcup_{\iota<\lambda} \mathcal{D}^\iota(R)\,.$$

4.2 When is one step enough?

Proving that (con-tot) holds for $\mathtt{CON}\langle\mathsf{con}_1\rangle$ and $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$ as it is, turned out to be too hard and, unless the authors have missed something obvious, we need to assume additional properties about the presentation. One of the reasons for the difficulty is the fact that $\mathtt{CON}\langle\mathsf{con}_1\rangle$ is computed as an iteration of $\mathcal{D}(-)$. In this subsection, we focus on the question whether there are natural properties, for a relation $R \subseteq L_+ \times L_-$, which guarantee $\mathcal{D}(\mathcal{D}(R)) = \mathcal{D}(R)$.

At the moment, R can be any relation on the frames but for the application to presentations we would like to instantiate R with $\downarrow\mathsf{con}_{\wedge,\vee}$. Because of that we will assume that R is downwards closed in $\sqsubseteq$-order and that it is closed under $\wedge$ and $\vee$.

We start with an important definition. Two sets $A_+ \subseteq L_+$ and $A_- \subseteq L_-$ are said to be *R-independent* if $\forall a_+ \in A_+$ and $\forall a_- \in A_-$, $(a_+, a_-) \in R$.

▶ **Observation 7.** *For every $\alpha \in R$, the sets $\mathcal{B}_+(\alpha_+)$ and $\mathcal{B}_-(\alpha_-)$ are R-independent where $\mathcal{B}_\pm(\alpha_\pm) \stackrel{def}{\equiv} \downarrow\alpha_\pm \cap B_\pm$.*

It turns out that $\mathcal{D}(R)$ can reformulated by using R-independent sets. Let $\alpha \in \mathcal{D}(R)$. From the definition, there is some directed $A \subseteq^\uparrow R$ such that $\alpha = \bigsqcup^\uparrow A$. Because $\mathcal{B}_\pm(-)$ are monotone and A is directed, the sets $\{\mathcal{B}_+(\alpha_+) : \alpha \in A\}$ and $\{\mathcal{B}_-(\alpha_-) : \alpha \in A\}$ are both also directed (in the subset order) and so we have:

$$\forall A \subseteq^\uparrow R \implies \bigcup_{\alpha\in A} \mathcal{B}_+(\alpha_+) \text{ and } \bigcup_{\alpha\in A} \mathcal{B}_-(\alpha_-) \text{ are } R\text{-independent} \qquad (\star)$$

Moreover, because $L_\pm$ is generated by $B_\pm$ and every $x \in L_\pm$ is equal to $\bigvee \mathcal{B}_\pm(x)$, we obtain that $\alpha = (\bigvee^\uparrow_{\alpha\in A} \alpha_+, \bigvee^\uparrow_{\alpha\in A} \alpha_-) = (\bigvee \mathcal{A}_+, \bigvee \mathcal{A}_-)$ where $\mathcal{A}_\pm = \bigcup_{\alpha\in A} \mathcal{B}_\pm(\alpha_\pm)$.

It might seem that $\mathcal{D}(-)$ is just a special case of a more general construction:

$$\mathcal{D}^{\mathrm{ind}}(R) = \{(\bigvee A_+, \bigvee A_-) \mid A_+ \subseteq B_+,\ A_- \subseteq B_- \text{ s.t. } A_+ \text{ and } A_- \text{ are } R\text{-independent}\}$$

What we have proved in the previous paragraphs is that $\mathcal{D}(R) \subseteq \mathcal{D}^{\mathrm{ind}}(R)$. In fact, both closures are equivalent:

▶ **Lemma 8.** $\mathcal{D}(R) = \mathcal{D}^{ind}(R)$

Proof. Only the right-to-left inclusion remains to be proved. Let $A_+ \subseteq B_+$ and $A_- \subseteq B_-$ be R-independent. Observe that for two finite sets $F_+ \subseteq^{\mathrm{fin}} A_+$ and $F_- \subseteq^{\mathrm{fin}} A_-$, $(\bigvee F_+, \bigvee F_-) \in R$. This is because R is $\vee$-closed and so $(\bigvee F_+, f_-) \in R$ for every $f_- \in F_-$ and, because R is $\wedge$-closed, $(\bigvee F_+, \bigvee F_-) \in R$. Clearly, the set $\mathcal{A} = \{(\bigvee F_+, \bigvee F_-) : F_+ \subseteq^{\mathrm{fin}} A_+ \text{ and } F_- \subseteq^{\mathrm{fin}} A_-\}$ is a directed subset of R and $(\bigvee A_+, \bigvee A_-) = \bigsqcup^\uparrow \mathcal{A} \in \mathcal{D}(R)$. ◀

Because $\mathcal{D}(R)$ is also downwards closed and closed under $\wedge$ and $\vee$ (Lemma 5), $\mathcal{D}(\mathcal{D}(R)) = \mathcal{D}^{\mathrm{ind}}(\mathcal{D}^{\mathrm{ind}}(R))$ and it might seem that this is already equal to $\mathcal{D}^{\mathrm{ind}}(R)$. But, this is not true in general. Take, for example, $\mathcal{A}_+ = \{a_+\}$ and $\mathcal{A}_- = \{a^1_-, a^2_-\}$ which are $\mathcal{D}^{\mathrm{ind}}(R)$-independent. Each of (a_+, a^1_-) and $(a_+, a^2_-) \in \mathcal{D}^{\mathrm{ind}}(R)$ is witnessed by a pair of R-independent sets A^1_+ and A^1_-, and A^2_+ and A^2_-, respectively, such that $a_+ = \bigvee A^1_+ = \bigvee A^2_+$ and $a^1_- = \bigvee A^1_-$ and $a^2_- = \bigvee A^2_-$. However, because there is no reason to believe that A^1_+ and A^2_+ are equal, there are no obvious candidates for R-independent sets which would have $(a_+, a^1_- \vee a^2_-)$ as their supremum. To overcome this problem, we assume the following condition:

(R-ind) For all $\forall \alpha \in \mathcal{D}^{\mathrm{ind}}(R)$, $\mathcal{B}_+(\alpha_+)$ and $\mathcal{B}_-(\alpha_-)$ are R-independent.

This guarantees, for every $\alpha \in \mathcal{D}^{\mathrm{ind}}(R)$, a canonical choice of R-independent sets, namely $A_\pm = \mathcal{B}_\pm(\alpha_\pm)$.

▶ **Lemma 9.** $\mathcal{D}(\mathcal{D}^{ind}(R)) \subseteq \mathcal{D}^{ind}(R)$

Proof. Let $A \subseteq^{\uparrow} \mathcal{D}^{\mathrm{ind}}(R)$. By ($R$-ind), for every $\alpha \in A$, $\mathcal{B}_+(\alpha_+)$ and $\mathcal{B}_-(\alpha_-)$ are R-independent. As in ($\star$), because A is directed, the sets $\mathcal{A}_+ \overset{\mathrm{def}}{\equiv} \bigcup^{\uparrow}_{\alpha\in A} \mathcal{B}_+(\alpha_+)$ and $\mathcal{A}_- \overset{\mathrm{def}}{\equiv} \bigcup^{\uparrow}_{\alpha\in A} \mathcal{B}_+(\alpha_+)$ are R-independent and $\bigsqcup^{\uparrow} A = (\bigvee \mathcal{A}_+, \bigvee \mathcal{A}_-)$. Hence, $\bigsqcup^{\uparrow} A \in \mathcal{D}^{\mathrm{ind}}(R)$. ◀

A combination of the preceding lemmas yields the desired result:

▶ **Theorem 10.** *Let $R \subseteq L_+ \times L_-$ be downwards closed, closed under logical meets and joins. If (R-ind) is true for R, then $\mathcal{D}(\mathcal{D}(R)) = \mathcal{D}(R)$.*

Proof. $\mathcal{D}(\mathcal{D}(R)) \overset{(Lemma\ 8)}{=} \mathcal{D}(\mathcal{D}^{\mathrm{ind}}(R)) \overset{(Lemma\ 9)}{\subseteq} \mathcal{D}^{\mathrm{ind}}(R) \overset{(Lemma\ 8)}{=} \mathcal{D}(R) \subseteq \mathcal{D}(\mathcal{D}(R))$ ◀

▶ Remark. Because $\mathcal{D}(R) = \mathcal{D}^{\mathrm{ind}}(R)$ is downwards closed, for every $\alpha \in \mathcal{D}(R)$ and every $(b_+, b_-) \in \mathcal{B}_+(\alpha_+) \times \mathcal{B}_-(\alpha_-)$, also $(b_+, b_-) \in \mathcal{D}(R)$. Therefore, ($R$-ind) can be reformulated in the following more compact way:

(R-ind) $(B_+ \times B_-) \cap \mathcal{D}(R) \subseteq R$

4.3 Chasing down (con-tot)

Finally, we can focus on the original (con-tot) axiom for $(L_+, L_-; \mathtt{CON}\langle \mathsf{con}_1 \rangle, \mathtt{TOT}\langle \mathsf{tot}_1 \rangle)$. We split it into two parts:

(λ^0_+-con-tot) $\alpha \in \mathtt{CON}\langle \mathsf{con}_1 \rangle$, $\beta \in \mathtt{TOT}\langle \mathsf{tot}_1 \rangle$ and $\alpha_+ = \beta_+ \implies \alpha_- \leq \beta_-$

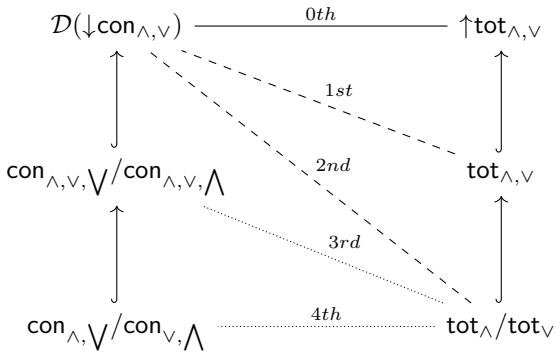
(λ^0_--con-tot) $\alpha \in \mathtt{CON}\langle \mathsf{con}_1 \rangle$, $\beta \in \mathtt{TOT}\langle \mathsf{tot}_1 \rangle$ and $\alpha_- = \beta_- \implies \alpha_+ \leq \beta_+$

If we assume (R-ind) about $\downarrow \mathsf{con}_{\wedge,\vee}$, then the conditions of Theorem 10 hold for $R = \downarrow \mathsf{con}_{\wedge,\vee}$ and we can rewrite ($\lambda^0_\pm$-con-tot) into the following more explicit form

(λ^0_+-con-tot) $\alpha \in \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee})$, $\beta \in \uparrow \mathsf{tot}_{\wedge,\vee}$ and $\alpha_+ = \beta_+ \implies \alpha_- \leq \beta_-$

(λ^0_--con-tot) $\alpha \in \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee})$, $\beta \in \uparrow \mathsf{tot}_{\wedge,\vee}$ and $\alpha_- = \beta_- \implies \alpha_+ \leq \beta_+$

Our aim now is to restrict α and β to smaller and smaller sets. First, we restate the axioms such that the β's come from $\mathsf{tot}_{\wedge,\vee}$ and then from $\mathsf{tot}_\wedge$ (resp. $\mathsf{tot}_\vee$). Then, we do the same with α until we obtain a version of the (con-tot) axiom stated purely in terms of formulas involving only elements from $\mathsf{con}_{\wedge,\bigvee}$ (resp. $\mathsf{con}_{\vee,\bigwedge}$) and $\mathsf{tot}_\wedge$ (resp. $\mathsf{tot}_\vee$). The individual stages are depicted in the diagram below (the λ superscripts in the axiom name correspond to the stages as shown in the diagram):



In every stage we introduce a pair of axioms (named ($\lambda^i_\pm$-con-tot), for $i = 1, \dots, 4$) and show that they imply the previous axioms. Because the axioms (λ^i_+-con-tot) and (λ^i_--con-tot) are dual to each other, we will always only prove that, say, (λ^i_+-con-tot) implies (λ^{i-1}_+-con-tot) and leave out that (λ^i_--con-tot) implies (λ^{i-1}_--con-tot) as it is proved dually.

▶ Remark. Above we use a notation similar to the one introduced earlier. The relation $\mathsf{con}_\vee$ is the algebraic closure of con_1 under all *finite* logical joins ($\vee$) in $L_+ \times L_-$, and $\mathsf{con}_\wedge$, $\mathsf{tot}_\vee$, $\mathsf{tot}_\wedge$, $\mathsf{tot}_{\wedge,\vee}$ and $\mathsf{con}_{\wedge,\vee}$ are defined correspondingly. Likewise, $\mathsf{con}_{\wedge,\bigvee}$ is the closure of con_1 under finite meets followed by the closure under all joins, both in logical order[5], i.e.

$$\mathsf{con}_{\wedge,\bigvee} = \{(\bigvee_i \alpha^i_+,\ \bigwedge_i \alpha^i_-) : \{\alpha^i\}_i \subseteq \mathsf{con}_\wedge\}.$$

The other versions, such as $\mathsf{con}_{\vee,\bigwedge}$, $\mathsf{con}_{\wedge,\vee,\bigvee}$ and $\mathsf{con}_{\wedge,\vee,\bigwedge}$, are defined correspondingly.

1st stage.

We intend to simplify the elements in the tot relation. Consider the following axioms:
(λ^1_+-con-tot) $\alpha \in \mathcal{D}(\downarrow\mathsf{con}_{\wedge,\vee})$, $\beta \in \mathsf{tot}_{\wedge,\vee}$, $\beta_+ \leq \alpha_+ \implies \alpha_- \leq \beta_-$
(λ^1_--con-tot) $\alpha \in \mathcal{D}(\downarrow\mathsf{con}_{\wedge,\vee})$, $\beta \in \mathsf{tot}_{\wedge,\vee}$, $\beta_- \leq \alpha_- \implies \alpha_+ \leq \beta_+$
Now, let $\alpha \in \mathcal{D}(\downarrow\mathsf{con}_{\wedge,\vee})$ and let $\beta \in \mathtt{TOT}\langle\mathsf{tot}_1\rangle$ with $\alpha_+ = \beta_+$. That means that there is some $\beta' \in \mathsf{tot}_{\wedge,\vee}$ such that $\beta' \sqsubseteq \beta$. We have that $\beta'_+ \leq \alpha_+$ and so we can now apply (λ^1_+-con-tot) and get that $\alpha_- \leq \beta'_-$ and so $\alpha_- \leq \beta'_- \leq \beta_-$. To sum up, we have proved the first part of:

▶ **Lemma 11.** *($\lambda^1_\pm$-con-tot) implies ($\lambda^0_\pm$-con-tot), and vice versa.*

For the converse assume $\beta_+ \leq \alpha_+$. Then the pair (α_+, β_-) belongs to $\uparrow\mathsf{tot}_{\wedge,\vee}$ and by ($\lambda^0_\pm$-con-tot) we can conclude $\alpha_- \leq \beta_-$.

2nd stage.

We can simplify the elements in tot even further. Take the axioms:
(λ^2_+-con-tot) $\alpha \in \mathcal{D}(\downarrow\mathsf{con}_{\wedge,\vee})$, $\beta \in \mathsf{tot}_{\wedge}$, $\beta_+ \leq \alpha_+ \implies \alpha_- \leq \beta_-$
(λ^2_--con-tot) $\alpha \in \mathcal{D}(\downarrow\mathsf{con}_{\wedge,\vee})$, $\beta \in \mathsf{tot}_{\vee}$, $\beta_- \leq \alpha_- \implies \alpha_+ \leq \beta_+$

[5] This makes sense because, in any d-frame $(L_+, L_-; \mathsf{con}, \mathsf{tot})$, $\{(\bigvee_i \alpha^i_+,\ \bigwedge_i \alpha^i_-) : \{\alpha^i\}_i \subseteq \mathsf{con}\} \subseteq \mathsf{con}$. Indeed, from (con–↓), all $(\alpha^i_+,\ \bigwedge_i \alpha^i_-) \in \mathsf{con}$ and, by $\vee$ and $\bigsqcup^\uparrow$closedness, $(\bigvee_i \alpha^i_+,\ \bigwedge_i \alpha^i_-) \in \mathsf{con}$.

Let $\alpha \in \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee})$ and let $\beta \in \mathsf{tot}_{\wedge,\vee}$ with $\alpha_+ \leq \beta_+$. We can decompose β such that $\beta = \bigvee_{k=1}^{n} \beta^k$ where $\beta^k \in \mathsf{tot}_\wedge$, for every $k = 1, \ldots, n$. Then, for every k, we have that $\beta^k_+ \leq \beta_+ \leq \alpha$ and so $\alpha_- \leq \beta^k_-$. Because $\alpha_- \leq \beta^k_-$ for every k, also $\alpha_- \leq \beta_- = \bigwedge_{k=1}^{n} \beta^k_-$.

▶ **Lemma 12.** *($\lambda^2_\pm$-con-tot) implies ($\lambda^1_\pm$-con-tot), and vice versa.*

Here the converse direction is trivial.

3rd stage.

Now we focus on the complexity of elements α from con. To eliminate $\mathcal{D}(-)$ consider the following auxiliary axioms:

(α_+-con-tot) $\{(x^k, y)\}_k \subseteq \downarrow \mathsf{con}_{\wedge,\vee}$, $\beta \in \mathsf{tot}_\wedge$, $\beta_+ \leq \bigvee_k x^k \implies y \leq \beta_-$

(α_--con-tot) $\{(x, y^k)\}_k \subseteq \downarrow \mathsf{con}_{\wedge,\vee}$, $\beta \in \mathsf{tot}_\vee$, $\beta_- \leq \bigvee_k y^k \implies x \leq \beta_+$

Let $\alpha \in \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee})$. By Lemma 8, this means that there exist $A_\pm \subseteq B_\pm$ which are $(\downarrow \mathsf{con}_{\wedge,\vee})$-independent and such that $\alpha = (\bigvee A_+, \bigvee A_-)$. Let us fix a $b_- \in A_-$. The $(\downarrow \mathsf{con}_{\wedge,\vee})$-independence of A_+ and A_- means that $A_+ \times \{b\} \subseteq \downarrow \mathsf{con}_{\wedge,\vee}$. Because also $\beta_+ \leq \alpha_+ = \bigvee A_+$, we can apply ($\alpha_+$-con-tot) and obtain that $b_- \leq \beta_-$. Since $b_- \in A_-$ was chosen arbitrarily, $\alpha_- = \bigvee A_- \leq \beta_-$. We have proved that ($\alpha_+$-con-tot) implies ($\lambda^2_+$-con-tot).

Finally, we can get rid of the downwards closure of $\mathsf{con}_{\wedge,\vee}$. Consider the following axioms:

(λ^3_+-con-tot) $\alpha \in \mathsf{con}_{\wedge,\vee,\bigvee}$, $\beta \in \mathsf{tot}_\wedge$, $\beta_+ \leq \alpha_+ \implies \alpha_- \leq \beta_-$

(λ^3_--con-tot) $\alpha \in \mathsf{con}_{\wedge,\vee,\bigwedge}$, $\beta \in \mathsf{tot}_\vee$, $\beta_- \leq \alpha_- \implies \alpha_+ \leq \beta_+$

Let $\{(x^k, y)\}_k \subseteq \downarrow \mathsf{con}_{\wedge,\vee}$ be such that $\beta_+ \leq \bigvee_k x^k$. For every k, there exists an $\alpha^k \in \mathsf{con}_{\wedge,\vee}$ such that $(x^k, y) \sqsubseteq \alpha^k$. Clearly, $\beta_+ \leq \bigvee_k x^k \leq \bigvee_k \alpha^k_+$, and $\alpha = (\bigvee_k \alpha^k_+, \bigwedge_k \alpha^k_-) \in \mathsf{con}_{\wedge,\vee,\bigvee}$. We can apply ($\lambda^3_+$-con-tot) and obtain that $y \leq \alpha_- \leq \beta_-$. Together with the previous result we have that:

▶ **Lemma 13.** *($\lambda^3_\pm$-con-tot) implies ($\lambda^2_\pm$-con-tot).*

4th stage.

The final simplification is similar to the 2nd stage but this time acts on the con side:

(λ^4_+-con-tot) $\alpha \in \mathsf{con}_{\wedge,\bigvee}$, $\beta \in \mathsf{tot}_\wedge$, $\beta_+ \leq \alpha_+ \implies \alpha_- \leq \beta_-$

(λ^4_--con-tot) $\alpha \in \mathsf{con}_{\vee,\bigwedge}$, $\beta \in \mathsf{tot}_\vee$, $\beta_- \leq \alpha_- \implies \alpha_+ \leq \beta_+$

Distributivity of $\wedge$ and $\vee$ gives us that

$$\mathsf{con}_{\wedge,\vee,\bigvee} = \mathsf{con}_{\wedge,\bigvee} \quad \text{and} \quad \mathsf{con}_{\wedge,\vee,\bigwedge} = \mathsf{con}_{\vee,\bigwedge}$$

from which we can conclude:

▶ **Lemma 14.** *($\lambda^4_\pm$-con-tot) implies ($\lambda^3_\pm$-con-tot), and vice versa.*

Furthermore, ($\lambda^4_\pm$-con-tot) and ($\lambda^1_\pm$-con-tot) are equivalent because

$$\mathsf{con}_{\wedge,\bigvee} \subseteq \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee}) \quad \text{and} \quad \mathsf{con}_{\vee,\bigwedge} \subseteq \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee}). \tag{2}$$

To prove these inclusions, let $\alpha = (\bigvee_k \alpha^k_+, \bigwedge_k \alpha^k_-)$ where $\{\alpha^k\}_{k \in K} \subseteq \mathsf{con}_\wedge$. Then, for every $k \in K$, $(\alpha^k_+, \alpha_-) \sqsubseteq \alpha^k$ and so $(\alpha^k_+, \alpha_-) \in \downarrow \mathsf{con}_\wedge \subseteq \downarrow \mathsf{con}_{\wedge,\vee}$. Because $\downarrow \mathsf{con}_{\wedge,\vee}$ is $\vee$-closed, $\{(\bigvee_{k \in F} \alpha^k_+, \alpha_-) : F \subseteq^{\text{fin}} K\}$ is directed in $\downarrow \mathsf{con}_{\wedge,\vee}$ and so $\alpha \in \mathcal{D}(\downarrow \mathsf{con}_{\wedge,\vee})$.

We can apply similar techniques to simplify (R-ind):

▶ **Lemma 15.** *($\downarrow\mathsf{con}_{\wedge,\vee}$-ind) is equivalent to having the following two conditions*
($\downarrow\mathsf{con}_{\wedge,\vee}$-ind$_+$) $(B_+ \times B_-) \cap \downarrow\mathsf{con}_{\wedge,\bigvee} \subseteq \downarrow\mathsf{con}_{\wedge,\vee}$
($\downarrow\mathsf{con}_{\wedge,\vee}$-ind$_-$) $(B_+ \times B_-) \cap \downarrow\mathsf{con}_{\vee,\bigwedge} \subseteq \downarrow\mathsf{con}_{\wedge,\vee}$

We sum up all the previous results into this theorem:

▶ **Theorem 16.** *If ($\lambda^4_\pm$-con-tot) and ($\downarrow\mathsf{con}_{\wedge,\vee}$-ind$_\pm$) hold for a pre-d-frame presentation, then the generated pre-d-frame satisfies (*con-tot*).*

▶ Remark. It does not seem possible to check if (con-tot) holds in the generated pre-d-frame just by looking at its syntactic presentation. However, our sufficient conditions are much simpler than the formulas involving infinitary applications of $\mathcal{D}(-)$. Nevertheless we still need to understand the structure of the generated frame components.

4.4 A special case

In our applications even stronger and simpler conditions hold for the presentations. Namely, consider the following *"micro version"* of (con-tot):
(μ_+-con-tot) $\alpha \in \mathsf{con}_\vee$, $\beta \in \mathsf{tot}_\wedge$, $\beta_+ \leq \alpha_+ \implies \alpha_- \leq \beta_-$
(μ_--con-tot) $\alpha \in \mathsf{con}_\wedge$, $\beta \in \mathsf{tot}_\vee$, $\beta_- \leq \alpha_- \implies \alpha_+ \leq \beta_+$
and the following (more powerful) version of conditions ($\downarrow\mathsf{con}_{\wedge,\vee}$-ind$_\pm$):
(Indep$_+$) $(L_+ \times B_-) \cap \downarrow\mathsf{con}_{\wedge,\bigvee} \subseteq \downarrow\mathsf{con}_\vee$
(Indep$_-$) $(B_+ \times L_-) \cap \downarrow\mathsf{con}_{\vee,\bigwedge} \subseteq \downarrow\mathsf{con}_\wedge$

▶ **Proposition 17.** *If ($\mu_\pm$-con-tot) and (Indep$_\pm$) hold for a pre-d-frame presentation, then the generated pre-d-frame satisfies (*con-tot*).*

5 Application: Coproducts

5.1 Coproducts of frames

For a nice presentation of the coproducts of frames, we refer the reader to the book *"Frames and Locales"* [12]. Here we only outline basic facts about the construction. Let $\{L^i\}_{i\in\mathcal{I}}$ be a family of frames. The coproduct of $\{L^i\}_i$ in the category of meet-semilattices is $\prod'_i L^i$ which is the subset of $\prod_i L^i$ consisting of those elements with all but finitely many coordinates equal to 1. Then, the coproduct of $\{L^i\}_i$ in the category of frames $\bigoplus_i L^i$ can be presented as the frame of $\mathcal{C}$-ideals of $(\prod'_i L^i, \mathcal{C})$ with the set of coverings $\mathcal{C}$ of the form:

$$\{a^k *_j u : k \in K\} \dashv (\bigvee_{k\in K} a^k) *_j u$$

where, for an $a \in L^j$ and $u \in \prod'_i L^i$, $a *_j u$ is the element of $\prod'_i L^i$ such that $(a *_j u)_j = a$ and $(a *_j u)_i = u_i$ for $i \neq j$. Recall also that the smallest element of $\bigoplus_i L^i$ is the $\mathcal{C}$-ideal $\mathbf{n} = \{u \in \prod'_i L^i \mid u_i = 0 \text{ for some } i\}$.

The inclusion maps are the frame homomorphisms $\iota^j : L^j \to \bigoplus_i L^i$, $x \mapsto \downarrow(x *_j \overline{1}) \cup \mathbf{n}$, where $(\overline{1})_i = 1$ for all $i \in \mathcal{I}$. We can factor ι^j into a composition of two meet-semilattice homomorphisms $[\![-]\!] \circ \kappa^j$ where

$$\kappa^j : L^j \to \prod'_i L^i \qquad \text{and} \qquad [\![-]\!] : \prod'_i L^i \to \bigoplus_i L^i$$
$$x \mapsto x *_j \overline{1} \qquad\qquad u \mapsto \downarrow u \cup \mathbf{n}$$

Here κ^j is the universal map for the semilattice coproduct $\prod'_i L^i$ and $[\![-]\!]$ is the inclusion $B \to \mathcal{C}\text{-}\mathtt{Idl}(B)$ as in Lemma 1, $B = \prod'_i L^i$, and $\downarrow u \cup \mathbf{n}$ is the smallest $\mathcal{C}$-ideal containing u.

5.2 Coproducts of d-frames

Let $\{\mathcal{L}^i = (L^i_+, L^i_-; \mathsf{con}^i, \mathsf{tot}^i)\}_{i\in\mathcal{I}}$ be a family of d-frames. We will define the coproduct of $\{\mathcal{L}^i\}_i$ by a free d-frame construction. First, we compute the frame components of the coproduct of $\{\mathcal{L}^i\}_i$ as the coproducts of the frame components of the d-frames $\{\mathcal{L}^i\}_i$. Set $B_+ = \prod'_i L^i_+$ and $B_- = \prod'_i L^i_-$ and $\mathcal{C}_+$ and $\mathcal{C}_-$ independently as in Subsection 5.1 (for B_+ and B_-, respectively). Namely, for every $j \in \mathcal{I}$, we have a frame homomorphism

$$\iota^j_\pm \colon L^j_\pm \xrightarrow{\kappa^j_\pm} \prod'_i L^i_\pm \xrightarrow{[\![-]\!]_\pm} \bigoplus_i L^i_\pm$$

In order for $\iota^j = (\iota^j_+, \iota^j_-)$ to be a *d-frame* embedding into a coproduct, for every $(a, b) \in \mathsf{con}^j$ (resp. tot^j), it has to be the case that $(\iota^j_+(a), \iota^j_-(b)) \in \mathtt{CON}\langle\mathsf{con}_1\rangle$ (resp. $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$). Also, the universal property of coproducts guarantees that for any d-frame cone $\{\mathcal{L}^i \to \mathcal{M}\}_i$ there is a mediating d-frame homomorphism $\bigoplus_i \mathcal{L}^i \to \mathcal{M}$. This means that the relations we generate $\mathtt{CON}\langle\mathsf{con}_1\rangle$ and $\mathtt{TOT}\langle\mathsf{tot}_1\rangle$ from should not contain anything more. Therefore, define $\mathsf{con}_1, \mathsf{tot}_1 \subseteq B_+ \times B_-$ by

$$\begin{aligned} (a *_j \overline{1},\, b *_j \overline{1}) \in \mathsf{con}_1 \quad &\text{iff} \quad (a, b) \in \mathsf{con}^j \\ (a *_j \overline{1},\, b *_j \overline{1}) \in \mathsf{tot}_1 \quad &\text{iff} \quad (a, b) \in \mathsf{tot}^j \end{aligned}$$

and by $\bigoplus_i \mathcal{L}^i$ denote the resulting pre-d-frame $(\bigoplus_i L^i_+ \times \bigoplus_i L^i_-;\, \mathtt{CON}\langle\mathsf{con}_1\rangle, \mathtt{TOT}\langle\mathsf{tot}_1\rangle)$.

▶ Notation. For every $a \in L^i$ and $u \in B$, denote $a \oplus_i u = [\![a *_i u]\!]_\pm = {\downarrow}(a *_i u) \cup \mathbf{n}_\pm$. In particular, $a \oplus_i \overline{1} = {\downarrow}(a *_i \overline{1}) \cup \mathbf{n}_\pm$. As before, we identify $B_\pm$ with $[\![B_\pm]\!]_\pm \subseteq \bigoplus_i L^i_\pm$, con_1 with $[\![\mathsf{con}_1]\!] \subseteq \bigoplus_i L^i_+ \times \bigoplus_i L^i_-$, and tot_1 with $[\![\mathsf{tot}_1]\!]$.

To simplify our work by making sure that we can deal with indexes coherently, we prove the following lemma about normal forms of elements from $\mathsf{con}_\vee$, $\mathsf{con}_\wedge$, $\mathsf{tot}_\vee$ and $\mathsf{tot}_\wedge$:

▶ **Lemma 18.** *Let $\alpha \in \mathsf{con}_\wedge/\mathsf{tot}_\wedge$. Then, it is of the form $(\bigwedge_i \alpha^i_+, \bigvee_i \alpha^i_-)$ such that*
1. *for every $i \in \mathcal{I}$: $\alpha^i = (a_+ \oplus_i \overline{1}, a_- \oplus_i \overline{1})$ for some $(a_+, a_-) \in \mathsf{con}^i$ (resp. tot^i), and*
2. *there exists a* finite *$I(\alpha) \subseteq^{fin} \mathcal{I}$ s.t. $i \in I(\alpha)$ iff $\alpha^i \neq \mathbb{tt}$*

Similarly, every $\alpha \in \mathsf{con}_\vee$ (resp. $\mathsf{tot}_\wedge$) is of the form $(\bigvee_i \alpha^i_+, \bigwedge_i \alpha^i_-)$ where $\alpha^i \in \mathsf{con}_1$ (resp. tot_1) and $I(\alpha)$ denotes the finite set of indexes for which $\alpha^i \neq \mathbb{ff}$.

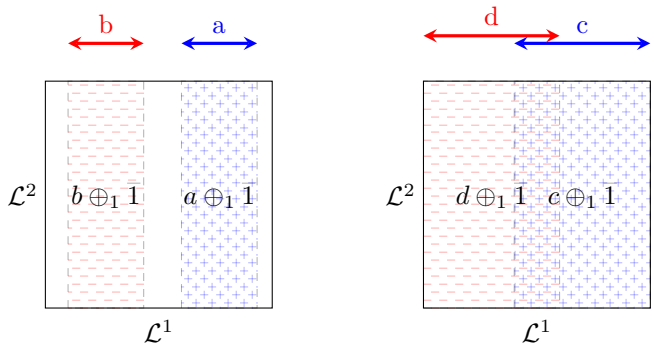
Notice that 1. and 2. make sense together. Anytime $\alpha^i = \mathbb{tt}$ we have that $\mathbb{tt} = ({\downarrow}\overline{1} \cup \mathbf{n}_+, \mathbf{n}_-) = (1 \oplus_i \overline{1}, 0 \oplus_i \overline{1})$ and $(1, 0) \in \mathsf{con}^i/\mathsf{tot}^i$. The case for $\alpha^i = \mathbb{ff}$ is similar.

5.3 Strips, rectangles and crosses

Before we get into proving that $\bigoplus_i \mathcal{L}^i$ satisfies (con-tot) we look into the structure of $\mathsf{con}_\vee$, $\mathsf{con}_\wedge$, $\mathsf{tot}_\vee$ and $\mathsf{tot}_\wedge$. It turns out that there is a nice geometrical intuition that we can employ.

First, for an $a \in L^i_\pm$, we call $a \oplus_i \overline{1}$ *an i-strip*[6]. Then, anytime $(a, b) \in \mathsf{con}^i$, we can think of the corresponding pair $(a \oplus_i \overline{1},\, b \oplus_i \overline{1}) \in \mathsf{con}_1$ as of a pair of *"disjoint" i*-strips and, similarly, $(c, d) \in \mathsf{tot}^i$ gives a pair of strips that are *"covering the whole space"*, i.e. $(c \oplus_i \overline{1},\, d \oplus_i \overline{1}) \in \mathsf{tot}_1$. This terminology is motivated by the case when $\mathcal{I} = \{1, 2\}$. Both cases are displayed in the picture below for $\mathcal{L}^1 \oplus \mathcal{L}^2$:

[6] We sometimes omit the index and call i-strips just strips whenever it does not lead to a confusion.

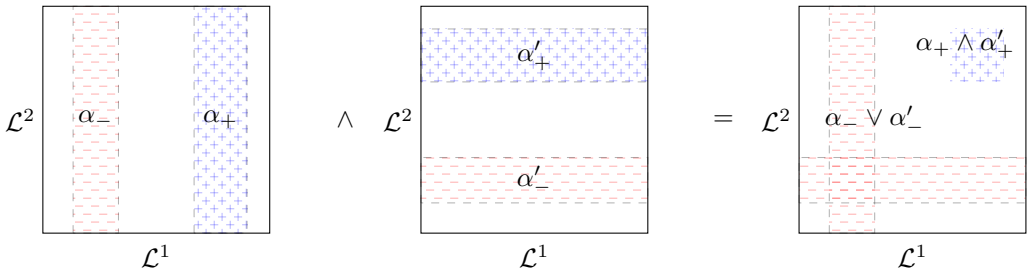


Therefore, all elements of con_1 and tot_1 are pairs of strips. It is rather a technical lemma that the set of i-strips in the coproduct has exactly the same structure as the d-frame $\mathcal{L}^i$:

▶ **Lemma 19.** *Let $S^i_\pm$ be the set of all i-strips in $\bigoplus_i L^i_\pm$. If all $L^i_\pm$'s are nontrivial*[7] *then*

$$(S^i_+, S^i_-;\, \mathsf{con}_1 \cap (S^i_+ \times S^i_-),\, \mathsf{tot}_1 \cap (S^i_+ \times S^i_-)) \cong \mathcal{L}^i.$$

Moreover, finite $\wedge$-combinations of pairs of strips is something that we can imagine as a pair consisting of a rectangle and a cross. For example, let $\alpha \in \mathsf{con}_1$ be a pair of 1-strips and $\alpha' \in \mathsf{con}_1$ a pair of 2-strips. Then, as the picture below suggests, the plus coordinate of $\alpha \wedge \alpha'$ in $\mathcal{L}^1 \oplus \mathcal{L}^2$ is a rectangle and the minus coordinate is a cross. Notice also that the cross and rectangle are disjoint.



The picture for two pairs of strips $\beta, \beta' \in \mathsf{tot}_1$ is similar but this time the cross and rectangle of $\beta \wedge \beta'$ cover the whole space.

This geometrical intuition builds up well for these formal definitions: $\gamma = \bigwedge_i \gamma^i$, where $\gamma^i = c^i \oplus_i \overline{1}$ $(\forall i \in \mathcal{I})$, is a *rectangle* if there exists a finite $I(\gamma) \subseteq^{\text{fin}} \mathcal{I}$ such that $c^i \neq 1$ iff $i \in I(\gamma)$. Similarly, $\delta = \bigvee_i \delta^i$, where $\delta^i = d^i \oplus_i \overline{1}$, is a *cross* if for some finite $I(\delta) \subseteq^{\text{fin}} \mathcal{I}$, $d^i \neq 0$ iff $i \in I(\delta)$.

Notice that, by Lemma 18, every element of $\mathsf{con}_\wedge$ (resp. $\mathsf{tot}_\wedge$) is of the form $(\bigwedge_i \alpha^i_+, \bigvee_i \alpha^i_-)$ with only finitely many nontrivial α^i's. In the present terminology, α is a pair rectangle–cross and this exactly matches the geometrical intuition we just discussed.

▶ **Observation 20.** *Rectangles are exactly the elements of $B_\pm$.*

Proof. Every $\gamma \in B_\pm$ is of the form $[\![u]\!]_\pm$ for some $u \in \prod'_i L^i_\pm$. Because u has only finitely many indexes different from 1, $[\![u]\!]_\pm = [\![(a^1 *_{i(1)} \overline{1}) \wedge \cdots \wedge (a^n *_{i(n)} \overline{1})]\!]_\pm = [\![a^1 *_{i(1)} \overline{1}]\!]_\pm \wedge \cdots \wedge [\![a^n *_{i(n)} \overline{1}]\!]_\pm = (a^1 \oplus_{i(1)} \overline{1}) \wedge \cdots \wedge (a^n \oplus_{i(n)} \overline{1})$. The reverse direction is similar. ◀

[7] A frame is *trivial* if it is isomorphic to the trivial frame $\mathbf{1} = \{0 = 1\}$.

There is a nice interplay between rectangles and crosses:

▶ **Lemma 21.** *Let $\gamma = \bigwedge_i \gamma^i$ be a rectangle and let $\delta = \bigvee_i \delta^i$ be a cross such that $\gamma \leq \delta$. Then, there exists an $i \in I(\gamma)$ such that $\gamma^i \leq \delta^i$.*

Proof. Let $\gamma^i = c^i \oplus_i \overline{1}$ and $\delta^i = d^i \oplus_i \overline{1}$, for every $i \in \mathcal{I}$. By Observation 20, $\gamma = [\![u]\!]_\pm$ for some $u \in \prod_i' L^i_\pm$ such that, for every $i \in \mathcal{I}$, $(u)_i = c^i$. This means that $(u)_i \neq 1$ iff $i \in I(\gamma)$. It is not difficult to check that δ has a form of a finite union $\bigcup_{i \in I(\delta)} \delta^i$. If $c^i = 0$ for some $i \in I(\gamma)$, then $c^i \leq d^i$. Otherwise, $c^i \neq 0$ for all $i \in I(\gamma)$ and, since $\gamma \leq \delta$ iff $u \in \delta$, there must exist an $i \in I(\delta)$ such that $u \in \delta^i$ and then $(u)_i = c^i \leq d^i$ (see Proposition 5.2 (4) in [12]). Finally, because $i \in I(\delta)$, $d^i \neq 1$ and so also $c^i \neq 1$ and $i \in I(\gamma)$. ◀

5.4 Proof of (con-tot)

In this section we prove that $\bigoplus_i \mathcal{L}^i$ is a d-frame. To simplify our proofs, we can assume that all $\mathcal{L}^i$'s are nontrivial thanks to the following lemma.

▶ **Lemma 22.** *If $L^i_+ = \mathbf{1}$ or $L^i_- = \mathbf{1}$ for some $i \in \mathcal{I}$, then $\bigoplus_i \mathcal{L}^i$ satisfies (con-tot).*

Proof. Observe that, by (con-tot) for $\mathcal{L}^i$, if $L^i_+ = \mathbf{1}$ then automatically also $L^i_- = \mathbf{1}$, and vice versa. Therefore, $\bigoplus_i L^i_\pm = \{\mathbf{n}_\pm\}$ and so $\bigoplus_i \mathcal{L}^i$ is trivial and satisfies (con-tot). ◀

To show that (con-tot) holds for $\bigoplus_i \mathcal{L}^i$ we will use Proposition 17. In order to be able to do that we need to prove that ($\mu_\pm$-con-tot) and (Indep$_\pm$) hold:

▶ **Lemma 23.** *($\mu_\pm$-con-tot) holds for $\bigoplus_i \mathcal{L}^i$:*

Proof. Let $\alpha = \bigvee_i \alpha^i \in \mathsf{con}_\vee$ and $\beta = \bigwedge_i \beta^i \in \mathsf{tot}_\wedge$ be in canonical forms, and assume that $\beta_+ \leq \alpha_+$. From canonicity of α and β, know that α_+ is a cross and β_+ is a rectangle. By Lemma 21, there is an $i \in I(\beta)$ such that $\beta^i_+ \leq \alpha^i_+$. From (con-tot) for $\mathcal{L}^i$, $\alpha^i_- \leq \beta^i_-$ and so $\alpha_- = \bigwedge_i \alpha^i_- \leq \alpha^i_- \leq \beta^i_- \leq \bigvee_i \beta^i_- = \beta_-$. ◀

▶ **Lemma 24.** *(Indep$_\pm$) holds for $\bigoplus_i \mathcal{L}^i$.*

Proof. Let $(x, b_-) \in (L_+ \times B_-) \cap {\downarrow}\mathsf{con}_{\wedge,\bigvee}$. Denote its upper bound $(\bigvee_k \alpha^k_+, \bigwedge_k \alpha^k_-)$ where, for each k, $\alpha^k = (\bigwedge_i \alpha^{k,i}_+, \bigvee_i \alpha^{k,i}_-)$ is a pair rectangle–cross from $\mathsf{con}_\wedge$. Because $b_- \in B_-$, it is a rectangle of the form $b_- = \bigwedge_i \gamma^i$ (Observation 20). Because, for every k, $b_- \leq \alpha^k_-$, by Lemma 21, there exists an $i(k) \in I(b_-)$ such that $\gamma^{i(k)} \leq \alpha^{k,i(k)}_-$. Fix an $i \in I(b_-)$ and set $K(i) = \{k \mid i(k) = i\}$. By Lemma 22, we can assume that all $L^i_\pm$'s are nontrivial and because $\{\alpha^{k,i} : k \in K(i)\}$ are all pairs of i-strips and γ^i is an i-strip, by Lemma 19, we can carry the reasoning in the rest of this paragraph in the d-frame $\mathcal{L}^i$. Since con^i is downwards closed and $\gamma^i \leq \alpha^{k,i}$ ($\forall k \in K(i)$), also $(\alpha^{k,i(k)}_+, \gamma^{i(k)}) \in \mathsf{con}_1$ and, therefore, by $\bigsqcup^\uparrow$and $\vee$-closeness of con^i, $(\bigvee_{k \in K(i)} \alpha^{k,i}_+, \gamma^i) \in \mathsf{con}_1$.

Finally, because $I(b_-)$ is finite

$$\bigvee_{i \in I(b_-)} \Big(\bigvee_{k \in K(i)} \alpha^{k,i}_+, \gamma^i \Big) = \Big(\bigvee_{i \in I(b_-)} \Big(\bigvee_{k \in K(i)} \alpha^{k,i}_+ \Big), \bigwedge_{i \in I(b_-)} \gamma^i \Big) = \Big(\bigvee_k \alpha^{k,i(k)}_+, b_- \Big) \in \mathsf{con}_\vee.$$

Because $\alpha^k_+ = \bigwedge_i \alpha^{k,i}_+ \leq \alpha^{k,i(k)}_+$ ($\forall k$), $x \leq \bigvee_k \alpha^k_+ \leq \bigvee_k \alpha^{k,i(k)}_+$ and so $(x, b_-) \in {\downarrow}\mathsf{con}_\vee$. ◀

By Proposition 17, we know that $\bigoplus_i \mathcal{L}^i$ is a d-frame and, moreover, by the same reasoning as for frames, we can prove that it has the universal property of a coproduct:

▶ **Theorem 25.** *$\bigoplus_i \mathcal{L}^i$ is the coproduct in the category of d-frames.*

Acknowledgements. Discussions with Aleš Pultr helped greatly in the simplifications of Section 4.

References

1. S. Abramsky. Domain theory in logical form. In *Symposium on Logic In Computer Science*, pages 47–53. IEEE Computer Society Press, 1987.
2. Richard N. Ball and Aleš Pultr. Extending semilattices to frames using sites and coverages. *Mathematica Slovaca*, 64(3):527–544, 2014.
3. B Banaschewski, GCL Brümmer, and KA Hardie. Biframes and bispaces. *Quaestiones Mathematicae*, 6(1-3):13–25, 1983.
4. Bart Jacobs. Many-sorted coalgebraic modal logic: a model-theoretic study. *RAIRO-Theoretical Informatics and Applications*, 35(1):31–59, 2001.
5. T. Jakl and A. Jung. Vietoris construction for bispaces and d-frames. In preparation.
6. P. T. Johnstone. *Stone Spaces*, volume 3 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1982.
7. A. Jung. Continuous domain theory in logical form. In B. Coecke, L. Ong, and P. Panangaden, editors, *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky*, volume 7860 of *Lecture Notes in Computer Science*, pages 166–177. Springer Verlag, 2013. doi:10.1007/978-3-642-38164-5_12.
8. A. Jung and M. A. Moshier. On the bitopological nature of Stone duality. Technical Report CSR-06-13, School of Computer Science, The University of Birmingham, 2006. 110 pages. URL: ftp://ftp.cs.bham.ac.uk/pub/tech-reports/2006/CSR-06-13.pdf.
9. Bartek Klin. Coalgebraic modal logic beyond sets. *Electronic Notes in Theoretical Computer Science*, 173:177–201, 2007.
10. Clemens Kupke, Alexander Kurz, and Yde Venema. Stone coalgebras. *Theoretical Computer Science*, 327(1-2):109–134, 2004.
11. Jimmie Lawson. Stably compact spaces. *Mathematical Structures in Computer Science*, 21(01):125–169, 2011.
12. Jorge Picado and Aleš Pultr. *Frames and Locales: Topology without points*. Springer-Birkhauser Basel, 2011.
13. S. J. Vickers. *Topology Via Logic*, volume 5 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

UML Interactions Meet State Machines – An Institutional Approach

Alexander Knapp[1] **and Till Mossakowski**[2]

1 Universität Augsburg, Germany
2 Otto-von-Guericke Universität Magdeburg, Germany

Abstract

UML allows the multi-viewpoint modelling of systems. One important question is whether an interaction as specified by a sequence diagram can be actually realised in the system. Here, the latter is specified as a combination of several state machines (one for each lifeline in the interaction) by a composite structure diagram. In order to tackle this question, we formalise the involved UML diagram types as institutions, and their relations as institution (co)morphisms.

1998 ACM Subject Classification D.2.1 Requirements/Specifications, D.2.2 Design Tools and Techniques

Keywords and phrases UML, state machines, interactions, composite structure diagrams, institutions, multi-view consistency

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.15

1 Introduction

The "Unified Modeling Language" (UML [17]) is a heterogeneous language: UML comprises a language family of 14 types of diagrams of structural and behavioural nature. These sub-languages are linked through a common meta-model, i.e., through abstract syntax; their semantics, however, is informally described mainly in isolation. In [10], we have outlined our research programme of "institutionalising UML". Our objective is to give, based on the theory of institutions [6], formal, heterogeneous semantics to UML, that — besides providing formal semantics for the individual sub-languages — ultimately allows to ask questions concerning the consistency between different diagram types and concerning refinement and implementation in a system development.

The horizontal dimension of the relationship between the different models has to ensure *consistency* of the models, i.e., that the models fit together and describe a coherent system. There are different kinds of consistency checks on the modelling level: Static checks ensuring type consistency and type correctness between types and instances. Dynamic checks include the properties and one or several cooperating instances or types. Most of the dynamic checks are theoretically undecidable, thus fully automatic tools will not be able to answer all instances. However, in many cases, useful automatic approximations are possible, while in other cases, manual effort may be involved.

In this paper, we study one such consistency problem that arises between UML state machine diagrams, UML composite structure diagrams, and UML sequence diagrams. The central question that we study is: Are the traces of an interaction diagram are realisable in a system of state machines, interlinked by a composite structure diagram? In order to answer this question, we need to define institutions for interaction diagrams and composite structure diagrams. These are also original contributions. By contrast, we can rely on a previous institution of state machines given in [11].

The remainder of this paper is structured as follows: In Section 2, we provide some background on our goal of heterogeneous institution-based UML semantics and introduce a small example illustrating interactions, state machines and composite structures. In Section 3, we recall an institution for state machines. The original contribution of this paper starts in Section 4, where we define an institution for interactions. Section 5 provides an institution for composite structures. Section 6 provides institution (co)morphisms for the interplay among these institutions, and discusses the verification of our main property of feasibility of an interaction. Finally, in Section 7 we conclude with an outlook to future work.

1.1 ATM Example

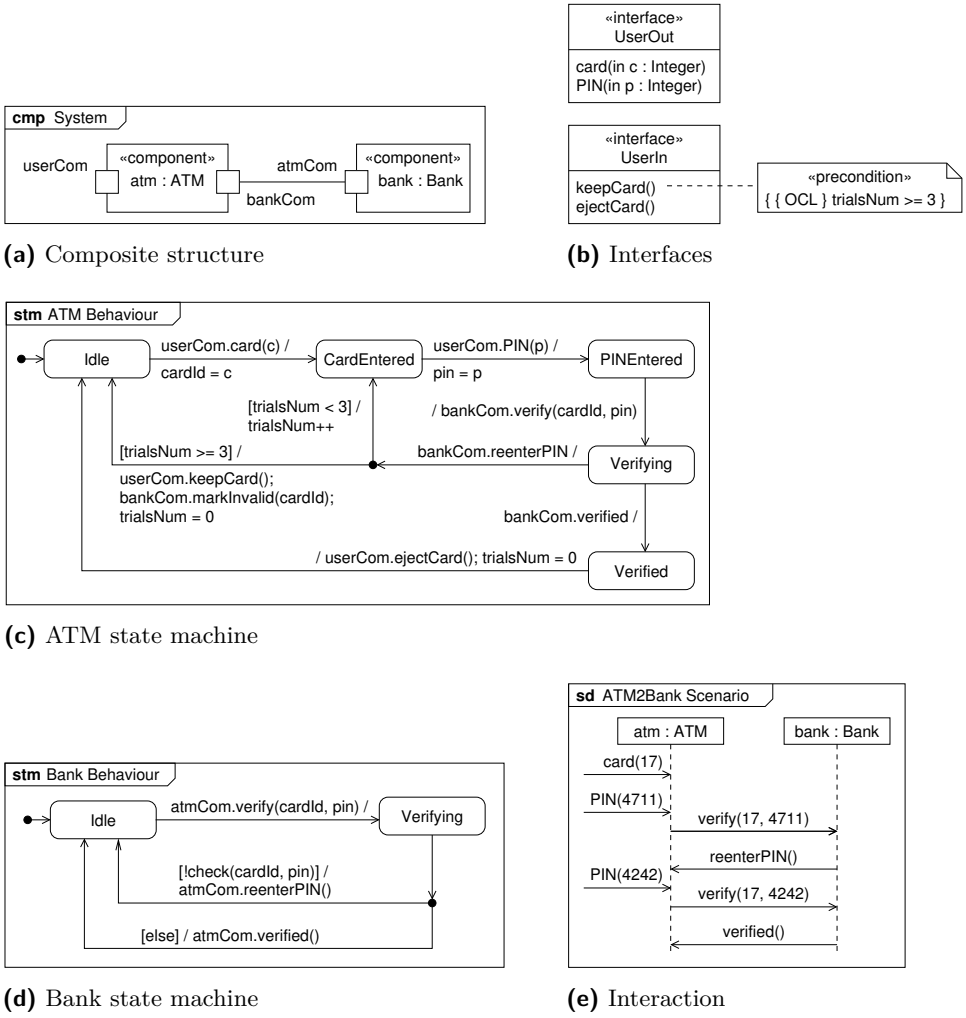
In order to illustrate our approach to a heterogeneous institutions-based UML semantics, we use as a small example the design of a traditional automatic teller machine (ATM) connected to a bank. For simplicity, we only describe the handling of entering a card and a PIN with the ATM. After entering the card, one has three trials for entering the correct PIN (which is checked by the bank). After three unsuccessful trials the card is kept.

Figure 1e shows a possible interaction between an atm and a bank in an environment as a UML sequence diagram which consists of seven messages: after receiving a card and a PIN input from the environment, the atm requests the bank to verify if the card and PIN number combination is valid; first, the bank requests to reenter the PIN, but after receiving a second PIN the verification is successful.

The composite structure of the ATM-bank system is specified in the *component diagram* in Figure 1a. In order to communicate with a bank component, the atm component has a *behaviour port* called bankCom and the bank component has a behaviour port atmCom. Furthermore, atm has a port userCom to a user. Interpreted at the component instance level this *composite structure diagram* also specifies the initial configuration of the system with the component instances atm and bank for the interaction.

Figure 1b provides structural information in the form of the interfaces specifying what is provided at the userCom port of the atm instance (UserIn) and what is required (UserOut). An interface is a set of operations that other model elements have to implement. In our case, the interface is described in a *class diagram*. Here, the operation keepCard is enriched with a pre-condition trialsNum >= 3 expressed in the "Object Constraint Language" (OCL) which refines its semantics: keepCard can only be invoked if the constraint holds.

The dynamic behaviour of the atm component is specified by the *behavioural state machine* shown in Figure 1c. The machine consists of five states including Idle, CardEntered, etc. Beginning in the initial Idle state, the user can *trigger* a state change by entering the card. This has the *effect* that the parameter c from the card event is assigned to cardId in the atm. Entering a PIN triggers another transition to PINEntered. Then the ATM requests verification from the bank using its bankCom port. The transition to Verifying uses a *completion event*: No explicit trigger is declared and the machine autonomously creates such an event whenever a state is completed, i.e., all internal activities of the state are finished (in our example there are no such activities). If the interaction with the bank results in reenterPIN, and the *guard* trialsNum < 3 is true, the user can again enter a PIN. In general, a state machine proceeds in *run-to-completion steps*: In a state, an event is fetched from the machine's event pool, where completion events are preferred; a transition outgoing from the state, triggered by the event, and with its guard satisfied is chosen; and the chosen transition is fired, leaving the source state, executing the transition's effects, and entering the target state. Message reception is recorded by an (external) event in the state machine's event pool.



(a) Composite structure

(b) Interfaces

(c) ATM state machine

(d) Bank state machine

(e) Interaction

Figure 1 ATM example

Another behavioural state machine specifies the behaviour of the bank component, see Figure 1d. For the sake of simplicity, we have omitted almost all details here. The machine waits in the starting Idle state until a verify event received via the atmCom port triggers the transition to the Verifying state. The machine then internally calls a check operation on the cardId and pin transmitted with the verify event. If this check fails, a reenterPIN event is sent to the ATM, otherwise, verified is sent. In reality, the machine and its check operation will be more involved. However, since these internals will not interfere with the ATM machine, they can be omitted here.

While this is a toy example, our envisaged interplay of interactions, state machines and composite structure diagrams is of industrial use: we cooperate with Fraunhofer IFF on applications of this very interplay to modelling parts of the smart electricity grid, e.g., adaptive grid protection devices and inter-station communication for voltage regulation.

2 Heterogeneous Institution-based UML Semantics

We have analysed in [9] that existing approaches to multi-view consistency in UML (i.e., addressing the question whether a family of UML diagrams is consistent) all have shortcomings and drawbacks. Indeed, there are only few approaches that cover more than three different diagram types, and very few that cover composite structure diagrams. Moreover, even those that do cover the latter do not provide means to answer our central question, whether the interaction expressed by the sequence diagram can be realised by the interplay of several state machines that are interconnected through a composite structure diagram.

This situation motivated us to start a larger effort [10] of giving an institution-based heterogeneous semantics to several UML diagrams. The specific choice of an institution-based approach is motivated in greater detail in [9]. The work in this paper is part of this effort. The vision is to provide semantic foundations for model-based specification and design using a heterogeneous framework based on Goguen's and Burstall's theory of institutions [6]. We handle the complexity of giving a coherent semantics to UML by providing several institutions formalising different diagrams of UML, and several institution translations (formalised as so-called institution morphisms and comorphisms) describing their interaction and information flow. The central advantage of this approach over previous approaches to formal semantics for UML (e.g., [13]) is that each UML diagram type can stay "as-is", without the immediate need of a coding using graph grammars (as in [3]) or some logic (as in [13]). Such coding can be done at verification time — this keeps full flexibility in the choice of verification mechanisms. The formalisation of UML diagrams as institutions has the additional benefit that a notion of refinement comes for free, see [2] and Section 6 below. Furthermore, the framework is flexible enough to support various development paradigms as well as different resolutions of UML's semantic variation points. This is the crucial advantage of the proposed approach to the semantics of UML, compared to existing approaches in the literature which map UML to a specific global semantic domain in a fixed way.

2.1 Institutions

Institutions are an abstract formalisation of the notion of logical systems. Informally, institutions provide four different logical notions: signatures, sentences, structures[1], and satisfaction. Signatures provide the vocabulary that may appear in sentences and that is interpreted in structures (= realisations). The satisfaction relation determines whether a given sentence is satisfied in a given structure. The exact nature of signatures, sentences, and structures is left unspecified, which leads to a great flexibility. This is crucial for the possibility to model UML diagrams (which in the first place are not "logics") as institutions.

More formally [6], an institution $\mathscr{I} = (\mathrm{Sig}^{\mathscr{I}}, \mathit{Sen}^{\mathscr{I}}, \mathrm{Str}^{\mathscr{I}}, \models^{\mathscr{I}})$ consists of (i) a category of *signatures* $\mathrm{Sig}^{\mathscr{I}}$; (ii) a *sentence functor* $\mathit{Sen}^{\mathscr{I}} : \mathrm{Sig}^{\mathscr{I}} \to \mathrm{Set}$, where Set is the category of sets; (iii) a contra-variant *structure functor* $\mathrm{Str}^{\mathscr{I}} : (\mathrm{Sig}^{\mathscr{I}})^{\mathrm{op}} \to \mathrm{Class}$, where Class is the category of classes; and (iv) a family of *satisfaction relations* $\models^{\mathscr{I}}_{\Sigma} \subseteq \mathrm{Str}^{\mathscr{I}}(\Sigma) \times \mathit{Sen}^{\mathscr{I}}(\Sigma)$ indexed over $\Sigma \in |\mathrm{Sig}^{\mathscr{I}}|$, such that the following *satisfaction condition* holds for every signature morphism $\sigma : \Sigma \to \Sigma'$ in $\mathrm{Sig}^{\mathscr{I}}$, every sentence $\varphi \in \mathit{Sen}^{\mathscr{I}}(\Sigma)$, and every Σ'-structure $M' \in \mathrm{Str}^{\mathscr{I}}(\Sigma')$:

$$\mathrm{Str}^{\mathscr{I}}(\sigma)(M') \models^{\mathscr{I}}_{\Sigma} \varphi \;\Leftrightarrow\; M' \models^{\mathscr{I}}_{\Sigma'} \mathit{Sen}^{\mathscr{I}}(\sigma)(\varphi) \;.$$

[1] Structures are called *models* in [6]. We use the term *structure* (and, interchangeably, *realisation*) here in order to avoid confusion with the term *model* in the sense of model-driven engineering.

$\mathrm{Str}^{\mathscr{I}}(\sigma)$ is called the *reduct* functor (also written $-|\sigma$), $Sen^{\mathscr{I}}(\sigma)$ the *translation* function (also written $\sigma(-)$).

A *theory* T in an institution consists of a signature Σ, written $sig(T)$, and a set of Σ-sentences; its structure class is the class of all Σ-structures satisfying the sentences.

In the next sections, we formalise different UML diagram types as institutions. We will also need to relate institutions via so-called comorphisms. They formalise the intuition of translating or encoding an institution into another one.

An *institution comorphism* [7] $\rho : \mathscr{I} \to \mathscr{J}$ consists of a functor $\Phi : \mathrm{Sig}^{\mathscr{I}} \to \mathrm{Sig}^{\mathscr{J}}$, a natural transformation $\alpha : Sen^{\mathscr{I}} \dot{\to} \Phi\,; Sen^{\mathscr{J}}$, and a natural transformation $\beta : \Phi^{\mathrm{op}}\,; \mathrm{Str}^{\mathscr{J}} \dot{\to} \mathrm{Str}^{\mathscr{I}}$, such that for any $\Sigma \in |\mathrm{Sig}^{\mathscr{I}}|$, for any $\varphi \in Sen^{\mathscr{I}}(\Sigma)$, and any $M' \in \mathrm{Str}^{\mathscr{J}}(\Phi(\Sigma))$ the following satisfaction condition holds:

$$M' \models^{\mathscr{J}}_{\Phi(\Sigma)} \alpha_\Sigma(\varphi) \;\Leftrightarrow\; \beta_\Sigma(M') \models^{\mathscr{I}}_{\Sigma} \varphi\ .$$

By contrast, institution morphisms formalise the intuition of mapping a "richer" institution onto a "poorer" or more abstract one. Both morphisms and comorphisms also come in a "semi" variant (i.e., semi-morphisms and semi-comorphisms) [7]. These omit both the sentence translation and the satisfaction condition. Semi-(co-)morphisms can provide a model-theoretic link between institutions that are too different to permit a sentence translation, e.g., interactions and state machines. Here, we only need semi-morphisms.

An *institution semi-morphism* $\mu : \mathscr{I} \to \mathscr{J}$ consists of a functor $\Phi : \mathrm{Sig}^{\mathscr{I}} \to \mathrm{Sig}^{\mathscr{J}}$ and a natural transformation $\beta : \mathrm{Str}^{\mathscr{I}} \dot{\to} \Phi^{\mathrm{op}}\,; \mathrm{Str}^{\mathscr{J}}$.

3 An Institution for Simple UML State Machines

We recall our institution for UML simple (non-hierarchical) state machines from [11]; an institution also covering hierarchical states has been developed in [5]. We only need the so-called flat state machine institution. *Signatures* Σ consist of a set of actions $A(\Sigma)$, a set of messages $M(\Sigma)$, a set of variables $V(\Sigma)$, a set of (external) events $E(\Sigma)$, a set of completion events $F(\Sigma)$, and a set of states $S(\Sigma)$, such that $E(\Sigma) \cap F(\Sigma) = \emptyset = E(\Sigma) \cap S(\Sigma)$. *Signature morphisms* map signatures component-wise, where the maps for $E(\Sigma)$, $F(\Sigma)$ and $S(\Sigma)$ need to be injective.

▶ **Example 1.** Considering the UML state machine ATM in Figure 1c, its signature *ATMSig* will contain the actions userCom.ejectCard(); trialsNum = 0 and trialsNum++, as well as the messages userCom.ejectCard() and bankCom.markInvalid(cardId). Variables will include trialsNum. There are no internal events. States (and completion events) will include Idle, CardEntered, PINEntered, Verifying and Verified. ◀

Sentences are labelled transition systems (LTS) with states in $S(\Sigma)$ and labels in $(E(\Sigma) \cup F(\Sigma)) \times G(V(\Sigma)) \times A(\Sigma) \times \wp(F(\Sigma))$, where $G(V(\Sigma))$ is a set of guard sentences over $V(\Sigma)$. For example, the graphs of the state machines in Figures 1c and 1d are such LTS.

▶ **Example 2.** Continuing the previous example for the state machine of Figure 1c defining the behaviour of ATM, this state machine can be represented as the following sentence over this signature:

$$(\mathsf{Idle}, \{\mathsf{Idle} \xrightarrow[T]{\mathsf{card(c)[true]/cardId = c,\emptyset}} \mathsf{CardEntered},$$
$$\mathsf{CardEntered} \xrightarrow[T]{\mathsf{PIN(p)[true]/pin = p,PINEntered}} \mathsf{PINEntered},$$

$$\mathsf{PINEntered} \xrightarrow[T]{\mathsf{PINEntered[true]/bank.verify(cardId,\ pin),\emptyset}} \mathsf{Verifying},$$
$$\mathsf{Verifying} \xrightarrow[T]{\mathsf{reenterPIN[trialsNum} < 3\mathsf{]/trialsNum{+}{+},\emptyset}} \mathsf{CardEntered}, \ldots\}) \ .$$

In particular, PINEntered occurs both as a state and as a completion event to which the third transition reacts. The junction pseudostate for making the decision whether trialsNum < 3 or trialsNum $>= 3$ has been resolved by combining the transitions. ◀

Structures involve a set of configurations $Conf(\Sigma)$. A configuration consists of a valuation over variables in $V(\Sigma)$ in some value domain, an event pool consisting of a sequence events over $E(\Sigma) \cup F(\Sigma)$, and a state $s \in S(\Sigma)$. Structures are pairs $\Theta = (I_\Theta, \Delta_\Theta)$ where $I_\Theta \subseteq Conf(\Sigma)$ is the set of initial configurations, and $\Delta_\Theta \subseteq Conf(\Sigma) \times Lbl(\Sigma) \times Conf(\Sigma)$ is a labelled transition relation, for a suitable set of labels Lbl. In [11], we use sets of messages from $M(\Sigma)$ as labels, i.e., $Lbl(\Sigma) = \wp(M(\Sigma))$. The drawback is that such a modelling of state machines does not cater for external events being consumed by the state machine. As a consequence, the interleaving product construction in [11] has to deeply look into state machine configurations. By contrast, we here want to treat configurations as black boxes. We therefore choose the set of labels as

$$Lbl(\Sigma) = \{out(M') \mid M' \subseteq M(\Sigma)\} \cup \{in(M') \mid M' \subseteq M(\Sigma)\} \ .$$

That is, a transition can either emit a set of messages M', in the same way as in [11]. Alternatively, a transition can also consume a set of messages M', which is then just added to the event pool of the machine's configuration (state and variable valuation are not affected).

Satisfaction of a sentence (which essentially is a syntactic LTS) in a structure (which essentially is a semantic LTS, i.e. enriches states with variable valuations and event pools) means that the semantic LTS makes moves as prescribed by the syntactic LTS, if the respective guard evaluates to true, while simultaneously updating the variable valuation and the event pool. For details, see [11]. Consumed messages $in(M')$ lead to a slight change w.r.t. [11], the formalisation of which is obvious.

Altogether, this gives us a flat state machine institution **SM**. Flatness here refers to the staged construction of the state machine institution in [11], starting with institutions of actions and guards. The flat construction in [11] directly incorporates these institutions into the state machine institution.

4 An Institution for Simple UML Interactions

UML sequence diagrams are one variant of UML interactions that specify the communication between several components (state machines and also users) of a system. We here formalise a simplified version of sequence diagrams as an institution.

Signatures Σ consist of a set $L(\Sigma)$ of *lifelines* and a set $M(\Sigma)$ of *messages*. For simplicity, we do not consider any typing of lifelines or messages. If Σ and Σ' are signatures, then a *signature morphism* $\sigma : \Sigma \to \Sigma'$ consists of an injective function $L(\sigma) : L(\Sigma) \to L(\Sigma')$ and a function $M(\sigma) : M(\Sigma) \to M(\Sigma')$.

Sentences over the signature Σ are given by the following grammar:

$$F ::= \mathsf{skip} \mid \mathsf{snd}(s, r, m) \mid \mathsf{snd}(s, m) \mid \mathsf{rcv}(s, r, m) \mid \mathsf{rcv}(r, m) \mid$$
$$\mathsf{strict}(F_1, F_2) \mid \mathsf{seq}(F_1, F_2) \mid \mathsf{par}(F_1, F_2) \mid \mathsf{alt}(F_1, F_2)$$

skip is the empty interaction. $\mathsf{snd}(s, r, m)$ denotes the event of lifeline $s \in L(\Sigma)$ sending message $m \in M(\Sigma)$ to lifeline $r \in L(\Sigma)$, whereas $\mathsf{snd}(s, m)$ denotes the sending from s to

the environment. Conversely, $\mathsf{rcv}(s, r, m)$ is the event of r receiving message m from s, and $\mathsf{rcv}(r, m)$ the reception of m from the environment. $\mathsf{strict}(F_1, F_2)$ is strict sequencing of interactions, i.e., all events in F_1 must occur before those in F_2. $\mathsf{seq}(F_1, F_2)$ is weak sequencing, only imposing the restriction that events keep their lifeline-wise order. $\mathsf{par}(F_1, F_2)$ allows for any parallel interleaving of F_1 and F_2. $\mathsf{alt}(F_1, F_2)$ chooses either F_1 or F_2. In this interaction sub-language we omit, in particular, guards, hiding by ignore and consider, and so-called negative behaviour as specified with neg and assert; a more comprehensive account is provided in [20]. Translation of a sentence by a signature morphism σ is the straightforward lifting of σ to the operators.

▶ **Example 3.** Consider the interaction in Figure 1e. Its signature *ScenarioSig* consists of two lifelines atm and bank, as well as the seven messages card(17), PIN(4711), verify(17, 4711), etc. The interaction is then directly expressed by the sentence

$$\begin{array}{l}\mathsf{seq}\big(\mathsf{rcv}(\mathsf{atm}, \mathsf{card}(17)),\\ \quad \mathsf{seq}\big(\mathsf{rcv}(\mathsf{atm}, \mathsf{PIN}(4711)),\\ \quad\quad \mathsf{seq}\big(\mathsf{strict}(\mathsf{snd}(\mathsf{atm}, \mathsf{bank}, \mathsf{verify}(17,\ 4711)), \mathsf{rcv}(\mathsf{atm}, \mathsf{bank}, \mathsf{verify}(17,\ 4711))),\\ \quad\quad\quad \mathsf{seq}\big(\mathsf{strict}(\mathsf{snd}(\mathsf{bank}, \mathsf{atm}, \mathsf{reenterPIN}()), \mathsf{rcv}(\mathsf{bank}, \mathsf{atm}, \mathsf{reenterPIN}())),\\ \quad\quad\quad\quad \mathsf{seq}\big(\mathsf{rcv}(\mathsf{atm}, \mathsf{PIN}(4242)), \ldots\big)\big)\big)\big)\big)\end{array}$$

where UML's default composition mechanism is weak sequencing, and strict sequencing is used to express UML's requirement that a message must be sent before it can be received. ◀

Structures over a signature Σ involve traces of events. *Events* $\mathscr{E}(\Sigma)$ are either of the form $snd(s, r, m)$ ("object $s \in L(\Sigma)$ sends invocation $m \in M(\Sigma)$ to object $r \in L(\Sigma)$"), $snd(s, m)$ ("s sends m to the environment"), $rcv(s, r, m)$ ("r receives m from s"), or $rcv(r, m)$ ("r receives m from the environment"). An *event trace* $t \in \mathscr{E}(\Sigma)^*$ is a sequence of events, where $\langle\rangle$ denotes the empty sequence and $e :: t$ event concatenation. A Σ-*structure* is then a set of event traces $T \subseteq \mathscr{E}(\Sigma)^*$. A signature morphism $\sigma : \Sigma \to \Sigma'$ is applied to events by $\sigma(snd(s, r, m)) = snd(L(\sigma)(s), L(\sigma)(r), M(\sigma)(m))$ and similarly for $\sigma(snd(s, m))$, $\sigma(rcv(s, r, m))$, and $\sigma(rcv(r, m))$; and also to event traces by $\sigma(\langle\rangle) = \langle\rangle$ and $\sigma(e :: t) = \sigma(e) :: \sigma(t)$. The *reduct* of $T' \subseteq \mathscr{E}(\Sigma')^*$ along $\sigma : \Sigma \to \Sigma'$ is defined by taking the preimage of σ on event traces:

$$T'|\sigma = \{t \mid \sigma(t) \in T'\} \ .$$

In order to define the semantics of sentences, we need some auxiliary notions on events and traces: The set of lifelines $\alpha(e)$ *active* in an event e is defined as $\alpha(snd(s, r, m)) = \{s\} = \alpha(snd(s, m))$ and $\alpha(rcv(s, r, m)) = \{r\} = \alpha(rcv(r, m))$. Two events e_1 and e_2 are *in conflict*, written as $e_1 \asymp e_2$, if they share active lifelines, i.e., $\alpha(e_1) \cap \alpha(e_2) \neq \emptyset$. We use the following binary operations on traces yielding sets of traces, which we will also apply to sets of traces defining $T_1 \diamond T_2 = \bigcup\{t_1 \diamond t_2 \mid t_1 \in T_1,\ t_2 \in T_2\}$:

Strict sequencing In $t_1 \,;\, t_2$, all events of t_1 must occur before all events of t_2:

$$\langle\rangle \,;\, t_2 = \{t_2\} \ , \quad (e :: t_1) \,;\, t_2 = \{e :: t \mid t \in t_1 \,;\, t_2\}$$

Weak sequencing In $t_1 \,;_{\not\asymp}\, t_2$, events of t_1 may also occur after events of t_2. However, the ordering on lifelines must be preserved; this is ensured by the conflict condition $\neg(e_1 \not\asymp e_2)$:

$$\begin{aligned}&\langle\rangle \,;_{\not\asymp}\, t_2 = \{t_2\} \ , \qquad t_1 \,;_{\not\asymp}\, \langle\rangle = \{t_1\} \ ,\\ &(e_1 :: t_1) \,;_{\not\asymp}\, (e_2 :: t_2) = \{e_1 :: t \mid t \in t_1 \,;_{\not\asymp}\, (e_2 :: t_2)\} \cup\\ &\qquad\qquad \{e_2 :: t \mid t \in (e_1 :: t_1) \,;_{\not\asymp}\, t_2,\ \neg(e_1 \not\asymp e_2)\}\end{aligned}$$

Interleaving $t_1 \parallel t_2$ implements parallel interleaving of traces t_1 and t_2:

$$\langle\rangle \parallel t_2 = \{t_2\} \;, \qquad t_1 \parallel \langle\rangle = \{t_1\} \;,$$
$$(e_1 :: t_1) \parallel (e_2 :: t_2) = \{e_1 :: t \mid t \in t_1 \parallel (e_2 :: t_2)\} \cup \{e_2 :: t \mid t \in (e_1 :: t_1) \parallel t_2\}$$

With these preliminaries, we are now ready to define the set of traces of a sentence:

$$\begin{aligned}
&\mathscr{P}(\mathsf{skip}) = \{\langle\rangle\} && \mathscr{P}(\mathsf{strict}(F_1, F_2)) = \mathscr{P}(F_1)\,;\,\mathscr{P}(F_2)\\
&\mathscr{P}(\mathsf{snd}(s, r, m)) = \{\langle snd(s, r, m)\rangle\} && \mathscr{P}(\mathsf{seq}(F_1, F_2)) = \mathscr{P}(F_1)\,;_{\lessapprox}\,\mathscr{P}(F_2)\\
&\mathscr{P}(\mathsf{snd}(s, m)) = \{\langle snd(s, m)\rangle\} && \mathscr{P}(\mathsf{par}(F_1, F_2)) = \mathscr{P}(F_1) \parallel \mathscr{P}(F_2)\\
&\mathscr{P}(\mathsf{rcv}(s, r, m)) = \{\langle rcv(s, r, m)\rangle\} && \mathscr{P}(\mathsf{alt}(F_1, F_2)) = \mathscr{P}(F_1) \cup \mathscr{P}(F_2)\\
&\mathscr{P}(\mathsf{rcv}(r, m)) = \{\langle rcv(r, m)\rangle\}
\end{aligned}$$

The *satisfaction relation* requires that a sentence is satisfied in a structure iff some of the sentence's traces occur in the structure:

$$T \models_\Sigma F \;\Leftrightarrow\; \mathscr{P}(F) \cap T \neq \emptyset \;.$$

In fact, injectivity of $L(\sigma)$ in signature morphisms is needed for ensuring the satisfaction condition: Consider Σ and Σ' with $L(\Sigma) = \{l_1, l_2, l_3, l_4\}$, $M(\Sigma) = \{m\} = M(\Sigma')$, $L(\Sigma') = \{l, l_3, l_4\}$, and $\sigma : \Sigma \to \Sigma'$ mapping l_1 and l_2 to l and the rest identically. Let $T' = \{\langle snd(l, l_4, m), snd(l, l_3, m)\rangle\}$ and $F = \mathsf{seq}(\mathsf{snd}(l_1, l_3, m), \mathsf{snd}(l_2, l_4, m))$. Then $T'|\sigma \models_\Sigma F$ (witnessed by the trace $\langle snd(l_2, l_4, m), snd(l_1, l_3, m)\rangle$), but $T' \not\models_{\Sigma'} \sigma(F)$ as $snd(l, l_4, m) \not\lessapprox snd(l, l_3, m)$ prohibiting the event order to be changed in the sequential composition.

To show the satisfaction condition for the injective case, we need a lemma:

▶ **Lemma 4.** *Let $\sigma : \Sigma \to \Sigma'$ be a signature morphism.*
1. $\mathscr{P}(\sigma(F)) \subseteq \sigma(\mathscr{E}(\Sigma)^*)$.
2. $\mathscr{P}(\sigma(F)) = \sigma(\mathscr{P}(F))$.

Proof. **(1)** By a straightforward induction on the structure of F.
(2) By induction on the structure of F. We only show the cases of skip, snd, and seq:

$$\begin{aligned}
&\mathscr{P}(\sigma(\mathsf{skip})) = \mathscr{P}(\mathsf{skip}) = \{\langle\rangle\} = \sigma(\{\langle\rangle\}) = \sigma(\mathscr{P}(\mathsf{skip}))\;;\\
&\mathscr{P}(\sigma(\mathsf{snd}(l_s, l_r, m))) = \mathscr{P}(\mathsf{snd}(L(\sigma)(l_s), L(\sigma)(l_r), M(\sigma)(m))) =\\
&\qquad \{\langle snd(L(\sigma)(l_s), L(\sigma)(l_r), M(\sigma)(m))\rangle\} = \sigma(\mathscr{P}(\mathsf{snd}(l_s, l_r, m)))\;;\\
&\mathscr{P}(\sigma(\mathsf{seq}(F_1, F_2))) = \mathscr{P}(\mathsf{seq}(\sigma(F_1), \sigma(F_2))) = \mathscr{P}(\sigma(F_1))\,;_{\lessapprox}\,\mathscr{P}(\sigma(F_2)) =\\
&\qquad \sigma(\mathscr{P}(F_1))\,;_{\lessapprox}\,\sigma(\mathscr{P}(F_2)) = \sigma(\mathscr{P}(F_1)\,;_{\lessapprox}\,\mathscr{P}(F_2)) = \sigma(\mathscr{P}(\mathsf{seq}(F_1, F_2)))\;,
\end{aligned}$$

where injectivity of $L(\sigma)$ is needed for $\sigma(\mathscr{P}(F_1))\,;_{\lessapprox}\,\sigma(\mathscr{P}(F_2)) = \sigma(\mathscr{P}(F_1)\,;_{\lessapprox}\,\mathscr{P}(F_2))$. ◀

▶ **Proposition 5** (Satisfaction condition)**.** *Let $\sigma : \Sigma \to \Sigma'$ be a signature morphism and T' a structure over Σ'. Then $T'|\sigma \models_\Sigma F \Leftrightarrow T' \models_{\Sigma'} \sigma(F)$.*

Proof. We have $T'|\sigma \models_\Sigma F$ iff $\mathscr{P}(F) \cap \sigma^{-1}(T') \neq \emptyset$ iff (*) $\sigma(\mathscr{P}(F)) \cap \sigma(\sigma^{-1}(T')) \neq \emptyset$ iff (by Lemma 4(2)) $\mathscr{P}(\sigma(F)) \cap \sigma(\sigma^{-1}(T')) \neq \emptyset$ iff (**) $\mathscr{P}(\sigma(F)) \cap T' \neq \emptyset$ iff $T' \models_{\Sigma'} \sigma(F)$. Step (**) from left to right follows since $\sigma(\sigma^{-1}(T')) \subseteq T'$. For the converse direction, if $\mathscr{P}(\sigma(F)) \cap T' \neq \emptyset$, by Lemma 4(1), $\mathscr{P}(\sigma(F)) \cap \sigma(\mathscr{E}(\Sigma)^*) \cap T' \neq \emptyset$, hence $\mathscr{P}(\sigma(F)) \cap \sigma(\sigma^{-1}(T')) \neq \emptyset$. Step (*) from left to right is clear. From right to left, if $t_1 \in \mathscr{P}(F)$ and $t_2 \in \sigma^{-1}(T')$ with $\sigma(t_1) = \sigma(t_2)$, then $t_1 \in \mathscr{P}(F) \cap \sigma^{-1}(T')$. ◀

Altogether, this gives us an institution **SD** of sequence diagrams.

If one wants to drop the restriction of injectivity for signature morphisms, one could use so-called extended structures [21]. This would mean to start with signature morphisms without injectivity restriction, which only leads to a so-called pre-institution (i.e., an institution without satisfaction condition). From a pre-institution, one can build an institution of extended structures. An extended structure over a signature Σ is a pair (σ, M') where $\sigma : \Sigma \to \Sigma'$ is a (pre-institution, here: possibly non-injective) signature morphism and M' is a Σ'-structure. The institution of extended structures always enjoys the satisfaction condition.

5 An Institution for Simple UML Composite Structures

A UML composite structure diagram specifies the linkage of component instances communicating through ports over connectors interlinking ports. We formalise a simplified (non-hierarchical) variant of composite structures here. All connectors are binary and each component instance is equipped with a state machine for describing its behaviour.

A composite structure *signature* Σ consists of a set of *components* $C(\Sigma)$; a *state machine assignment* $S(\Sigma) : C(\Sigma) \to |\mathbf{SM}\text{-Sig}|$, where **SM**-Sig is the category of signatures of the flat state machine institution (see Section 3); a set $P(\Sigma)$ of *ports* p each showing a component $c(p) \in C(\Sigma)$, a port name $n(p)$, and a set of messages $M(p)$; and a symmetric binary relation $\Gamma(\Sigma) \subseteq P(\Sigma) \times P(\Sigma)$ of *connectors* that connect ports, such that

- for each port $p \in P(\Sigma)$ and each message $m \in M(p)$, the prefixed message $n(p).m$ is a message of component c's state machine, i.e., $n(p).m \in M(S(\Sigma)(c))$, and
- for each connector $(p_1, p_2) \in \Gamma(\Sigma)$, $M(p_1) = M(p_2)$, i.e., messages are fully compatible.

We say that port $p \in P(\Sigma)$ is *open* in $\Gamma(\Sigma)$ if there is no $p' \in P(\Sigma)$ such that $(p, p') \in \Gamma(\Sigma)$; otherwise p is *connected*.

Note that we consider the message components $M(S(\Sigma)(c))$ of the state machine signatures only. The event components are not relevant here, because they refer to internal events, and these cannot be sent over a connection.

▶ **Example 6.** Consider the composite structure in Figure 1a. Its signature *SystemSig* features two components atm and bank. The state machine signature $S(\mathit{SystemSig})(\mathsf{atm})$ is that of Example 1. The set of ports is $P(\mathit{SystemSig}) = \{\mathit{atmCom}, \mathit{bankCom}\}$ with $c(\mathit{bankCom}) = \mathsf{atm}$, $c(\mathit{atmCom}) = \mathsf{bank}$, $n(\mathit{atmCom}) = \mathsf{atmCom}$, $n(\mathit{bankCom}) = \mathsf{bankCom}$, and $M(\mathit{atmCom}) = \{\mathsf{verify}(-,-), \mathsf{reenterPIN}(), \mathsf{verified}(), \mathsf{markInvalid}(-)\} = M(\mathit{bankCom})$. The placeholders $-$ denote possible arguments, leading to an infinite set of possible messages. The set of connectors is $\Gamma(\mathit{SystemSig}) = \{(\mathit{atmCom}, \mathit{bankCom}), (\mathit{bankCom}, \mathit{atmCom})\}$.

The first condition on signatures means that the involved state machines must be able to communicate all port messages by prefixing with the port name. For example, the bank state machine in Figure 1d can emit a message atmCom.verified, and the ATM state machine in Figure 1c can receive a message bankCom.verified as trigger. Both messages are prefixed with the port name. Inside the port, the message is just verified in both cases. The second condition on signatures ensures that all such shared messages can be sent over the connector. ◀

A *signature morphism* $\sigma : \Sigma \to \Sigma'$ consists of a function $C(\sigma) : C(\Sigma) \to C(\Sigma')$ mapping components; for each $c \in C(\Sigma)$, a signature morphism $S(\sigma)(c) : S(\Sigma)(c) \to S(\Sigma')(C(\sigma)(c)) \in \mathbf{SM}\text{-Sig}$ interfacing the state machine signatures; a function $P(\sigma) : P(\Sigma) \to P(\Sigma')$ mapping ports; and a function $M(\sigma) : \bigcup\{M(p) \mid p \in P(\Sigma)\} \to \bigcup\{M(p') \mid p' \in P(\Sigma')\}$ mapping messages, such that

- for each port $p \in P(\Sigma)$, $M(\sigma)(M(p)) \subseteq M(P(\sigma)(p))$, that is, $M(\sigma)$ restricts to the port appropriately, and

- $(P(\sigma) \times P(\sigma))(\Gamma(\Sigma)) \subseteq \Gamma(\Sigma')$, i.e., connectors are preserved.

Sentences are pairs (c, φ) with $c \in C$ and $\varphi \in \mathbf{SM}\text{-Sen}(S(\Sigma)(c))$. Thus, on any component, we can impose state machine sentences over the signature of that component. Sentence translation is defined by $\sigma(c, \varphi) = (C(\sigma)(c), S(\sigma)(c)(\varphi))$.

Structures are families $R = (R(c) \in \mathbf{SM}\text{-Str}(S(\Sigma)(c)))_{c \in C(\Sigma)}$ of state machine structures. Given a signature morphism $\sigma : \Sigma \to \Sigma'$ and a Σ'-structure $R' = (R'(c') \in \mathbf{SM}\text{-Str}(S(\Sigma')(c')))_{c' \in C(\Sigma')}$, its *$\sigma$-reduct* is

$$R'|\sigma = (R'(C(\sigma)(c))|S(\sigma)(c))_{c \in C(\Sigma)} .$$

Satisfaction is defined by selecting the appropriate component:

$$R \models_\Sigma (c, \varphi) \;\Leftrightarrow\; R(c) \models^{\mathbf{SM}}_{S(\Sigma)(c)} \varphi .$$

Also, the satisfaction condition holds:

▶ **Proposition 7.** *Let $\sigma : \Sigma \to \Sigma'$ be a signature morphism and R' a structure over Σ'. Then $R'|\sigma \models_\Sigma (c, \varphi) \Leftrightarrow R' \models_{\Sigma'} \sigma(c, \varphi)$.*

Proof. $R'|\sigma \models_\Sigma (c, \varphi)$ iff $R'(C(\sigma)(c))|S(\sigma)(c) \models^{\mathbf{SM}}_{S(\Sigma)(c)} \varphi$ iff (by the satisfaction condition in **SM**) $R'(C(\sigma)(c)) \models^{\mathbf{SM}}_{S(\Sigma')(C(\sigma)(c))} S(\sigma)(c)(\varphi)$ iff $R' \models_{\Sigma'} \sigma(c, \varphi)$. ◀

Altogether, this gives us an institution **CMP** of composite structure diagrams.

Note that ports and connectors do not play a role for the satisfaction relation. They will play a role in the interaction with other institutions, the topic of the next section.

6 Putting It All Together

Our ultimate goal is the formalisation of the question whether the traces of an interaction diagram are realisable in a system of state machines, interlinked by a composite structure diagram. We will now introduce the institution translations necessary for reaching this goal.

6.1 Comorphism SM → CMP

We introduce a comorphism **SM** → **CMP** that can be used to construe a single state machine as a (singleton) composite structure. It is a trivial embedding and acts as follows:

Signatures $\Sigma_{\mathbf{SM}} \mapsto \Sigma_{\mathbf{CMP}}$ with $C(\Sigma_{\mathbf{CMP}}) = \{cid\}$, $S(\Sigma_{\mathbf{CMP}})(cid) = \Sigma_{\mathbf{SM}}$, $P(\Sigma_{\mathbf{CMP}}) = \emptyset$, and $\Gamma(\Sigma_{\mathbf{CMP}}) = \emptyset$

Sentences $\varphi_{\mathbf{SM}} \mapsto (cid, \varphi_{\mathbf{SM}})$

Realisations $R_{\mathbf{CMP}} \mapsto R_{\mathbf{CMP}}(cid)$

The satisfaction condition follows trivially.

Note that the resulting composite structure will name the single component *cid*. This can be changed with a renaming along a signature morphism.

6.2 Semi-morphism CMP → SD

Since interactions and composite structures have very different sentences, all we can hope for is to relate them via an institution semi-morphism **CMP** → **SD**. On signatures, it construes components as lifelines, and it assembles all possible messages that can be sent via the connectors. These are messages that are available in the state machines of the components, but prefixed with the respective port name. For example, the bank state machine in Figure 1d

can emit a message atmCom.verified, and the ATM state machine in Figure 1c can receive a message bankCom.verified as trigger. In the interaction of Figure 1e, there is a message verified that corresponds to the message atmCom.verified sent over the connector and being received as bankCom.verified. Hence, the message verified will appear in the resulting interaction signature.

More formally, a composite structure signature $\Sigma_{\mathbf{CMP}}$ is mapped to the interaction signature $\Phi(\Sigma_{\mathbf{CMP}})$ with $L(\Phi(\Sigma_{\mathbf{CMP}})) = C$ and

$$M(\Phi(\Sigma_{\mathbf{CMP}})) = \bigcup\{M(p) \mid p \in P(\Sigma_{\mathbf{CMP}})\}$$

A signature morphism $\sigma_{\mathbf{CMP}} : \Sigma_{\mathbf{CMP}} \to \Sigma'_{\mathbf{CMP}}$ is mapped to $\Phi(\sigma_{\mathbf{CMP}}) : \Phi(\Sigma_{\mathbf{CMP}}) \to \Phi(\Sigma'_{\mathbf{CMP}})$ with $L(\Phi(\sigma_{\mathbf{CMP}})) = C(\sigma_{\mathbf{CMP}})$ and $M(\Phi(\sigma_{\mathbf{CMP}})) = M(\sigma_{\mathbf{CMP}})$.

Concerning structures, note that a structure in **CMP** is a family of state machine structures: Form the interleaving product of these, and take the traces over the interleaved product. This is defined as a set of the message sequences of all possible runs in the labelled transition system, starting with an initial state. This gives an interaction structure.

We will now make this construction more precise. Note that we do not use the interleaved product construction from [11], because it is not abstract enough. That construction assumes that variable sets are shared (i.e., "shared memory"), and it moreover looks into the configurations of the state machines, e.g., in order to inject events into their event pools. By contrast, we here take a black-box view on state machines, and only use labels on transitions for communication among state machines. As a side effect, we also overcome the restriction made in [11] that state sets of the involved state machines must be disjoint, as well as their event sets. Moreover, a major difference is that in [11] we have used shared message names for communication, while here, communication happens only via explicit connectors.

Given a $\Sigma_{\mathbf{CMP}}$-structure R in **CMP**, we construct an LTS $\Pi(R)$ representing the interleaved product as follows: Let us abbreviate $C(\Sigma_{\mathbf{CMP}})$ by $C_{\mathbf{CMP}}$, $S(\Sigma_{\mathbf{CMP}})$ by $S_{\mathbf{CMP}}$, and $\Gamma(\Sigma_{\mathbf{CMP}})$ by $\Gamma_{\mathbf{CMP}}$. Let $R(c) = (I_{R_c}, \Delta_{R_c})$ for each $c \in C_{\mathbf{CMP}}$. The state space of $\Pi(R)$ is given by $\prod_{c \in C_{\mathbf{CMP}}} Conf(S_{\mathbf{CMP}}(c))$. Its set of initial states is given by $\prod_{c \in C_{\mathbf{CMP}}} I_{R_c}$. The transitions are given by

$$(s_c)_{c \in C_{\mathbf{CMP}}} \xrightarrow[\Pi(R)]{E} (s'_c)_{c \in C_{\mathbf{CMP}}}$$

if there is a component $\hat{c} \in C_{\mathbf{CMP}}$ such that

- either $s_{\hat{c}} \xrightarrow[\Delta_{R_{\hat{c}}}]{in(M_{\hat{c}})} s'_{\hat{c}}$ with all ports in $\{p \mid c(p) = \hat{c},\ n(p).m \in M_{\hat{c}}\}$ open in $\Gamma_{\mathbf{CMP}}$, $E = \{rcv(\hat{c}, m) \mid n.m \in M_{\hat{c}}\}$, and $s_c \xrightarrow[\Delta_{R_c}]{out(\emptyset)} s'_c$ or $s_c = s'_c$ for all $c \in C_{\mathbf{CMP}} \setminus \{\hat{c}\}$; i.e., component $\hat{c}$ receives input from the environment through its open ports and all other components make only internal steps;
- or $s_{\hat{c}} \xrightarrow[\Delta_{R_{\hat{c}}}]{out(M'_{\hat{c}})} s'_{\hat{c}}$ with $\{p \mid c(p) = \hat{c},\ n(p).m \in M_{\hat{c}}\} = P_0 \cup P_1$ such that all ports in P_0 are open and all ports in P_1 connected in $\Gamma_{\mathbf{CMP}}$, $E = \{snd(\hat{c}, m) \mid p_0 \in P_0,\ n(p_0).m \in M_{\hat{c}}\} \cup \{snd(\hat{c}, c(p_2), m), rcv(\hat{c}, c(p_2), m) \mid p_1 \in P_1,\ (p_1, p_2) \in \Gamma_{\mathbf{CMP}},\ n(p_1).m \in M'_{\hat{c}}\}$, $s_{c(p_2)} \xrightarrow[\Delta_{R_{c(p_2)}}]{in(M_{c(p_2)})} s'_{c(p_2)}$ with $M_{c(p_2)} = \{n(p_2).m \mid n(p_1).m \in M_{\hat{c}}\}$ for each $(p_1, p_2) \in \Gamma_{\mathbf{CMP}}$ and $p_1 \in P_1$, and $s_c \xrightarrow[\Delta_{R_c}]{out(\emptyset)} s'_c$ or $s_c = s'_c$ for $c \in C_{\mathbf{CMP}} \setminus (\{\hat{c}\} \cup \{c(p_2) \mid p_1 \in P_1,\ (p_1, p_2) \in \Gamma_{\mathbf{CMP}}\})$; i.e., component $\hat{c}$ sends messages which are received either by connected partners or the environment, and all other components not participating in the communication make only internal steps.

Based on this, the set of traces $Tr(\Pi(R))$ of $\Pi(R)$ is defined as follows: A *finite run* in $\Pi(R)$ is an alternating sequence $s_0\, E_1\, s_1\, \ldots\, E_n\, s_n$ with $s_0 \in I$ and $s_{i-1} \xrightarrow[\Pi(R)]{E_i} s_i$ for $i = 1, \ldots, n$. In this case, any $\overline{E}_1\, \ldots\, \overline{E}_n$ is a *trace* of $\Pi(R)$, where for $i = 1, \ldots, n$, $\overline{E}_i$ is a sequence with the same elements as the set E_i and all *snd*-events occur before all *rcv*-events.

$Tr(\Pi(R))$ is the translation of the structure R by the semi-morphism.

▶ **Example 8.** Consider the composite structure diagram in Figure 1a, showing instances atm and bank of the ATM and Bank components, respectively, that are connected through their bankCom and atmCom ports. In execution, atm and bank will exchange messages, as prescribed by their state machines, and this exchange is reflected by the interleaving product. Messages atmCom.verified (sent by the bank state machine) and bankCom.verified (received by the ATM state machine) will be unified to message verified in the interleaving product LTS and in its traces. ◀

6.3 Are Interactions Realisable?

Our initial question was whether the traces of an interaction diagram realisable in a system of state machines, interlinked by a composite structure diagram. This question can now be formalised as follows: The system of state machines is formalised as a theory in the institution **CMP**, using the comorphism $\mathbf{SM} \to \mathbf{CMP}$ to inject individual state machines into the system. This theory typically has a unique structure (because state machine theories have a unique structure). The structure of this theory in **CMP** can be translated along the semi-morphism $\mathbf{CMP} \to \mathbf{SD}$, resulting in a set of traces. If this intersects with the traces of the interaction, the interaction is realisable in the system of state machines.

This verification condition can also be expressed in the Distributed Ontology, Model and Specification Language (DOL) [18, 14][2], which recently has been adopted as a standard by the Object Management Group (OMG). UML diagrams can be referenced in DOL as-is using the standard interchange format XMI without the need of an encoding into some other language. The system of two state machines linked by a composite structure diagram can be expressed as follows, using the institution comorphism `sm2cmp` : $\mathbf{SM} \to \mathbf{CMP}$ introduced in Section 6.1 above:

```
model System =
  ATM_behaviour with translation sm2cmp with cid |-> atm
and
  Bank_behaviour with translation sm2cmp with cid |-> bank
then
  cmp
end
```

Recall that `sm2cmp` provides exactly one component with name `cid`. Using renaming in DOL, we can create two different components of the composite structure diagram, which are joined by a DOL union. The part `cmp` contains the specification of ports and connectors from the UML diagram.

We now express that this system can realise the interactions expressed in sequence diagram `ATM2Bank_Scenario` as a refinement in DOL:

```
refinement r2 =
  ATM2Bank_Scenario refined to { System hide along cmp2sd }
end
```

[2] See also `http://www.omg.org/spec/DOL/` and `http://dol-omg.org`.

Here, `cmp2sd` : **CMP** → **SD** is the institution semi-morphism introduced in Section 6.2. The semantics of the DOL refinement is just structure class inclusion. This means that each structure of `System hide along cmp2sd` must be a structure of `ATM2Bank_Scenario`. Now a structure of `System` is the (typically unique) family of semantic transition systems for the involved state machines (here `atm` and `bank`). `hide along cmp2sd` invokes translation along the institution semi-morphism `cmp2sd`. The result is the set of traces in the interleaved product. The requirement is now that this is a structure of `ATM2Bank_Scenario`, which means that at least one trace of the interleaved product must be a trace of `ATM2Bank_Scenario`. Hence, this exactly captures the condition that the interaction is realisable in the system of state machines as specified by the composite structure diagram.

▶ **Example 9.** The interaction in Figure 1e can be realised by the composite structure in Figure 1a that connects the state machines in Figure 1d and Figure 1c. ◀

During a typical development process, UML diagrams are refined and more details are added. We have

▶ **Proposition 10.** *Realisability of interactions is preserved when the composite structure diagram (with its state machines) is refined.*

Proof. By the above discussion, realisability of interactions can be expressed as a refinement. Moreover, refinements compose. ◀

Note that realisability of interactions is in general *not* preserved when the interaction itself is refined.

7 Conclusions

We have presented an important step in our program of formalising families of UML diagrams using institutions. This formalisation makes UML diagrams ready for use with the OMG-standardised Distributed Ontology, Model and Specification Language (DOL), which for example can express refinements. Our initial question whether interaction is realisable in the system of state machines as specified by the composite structure diagram can be indeed expressed as a DOL refinement. Such meta relations between UML models are normally expressed in an ad-hoc way. With our approach, meta relations can be expressed in DOL, a formal language with a formal, institution-based semantics.

Related approaches to semantics of UML composite structures are [19, 1, 16]. While [1] only consider static semantics, [19, 16] also consider behaviour. However, [19] do cover neither interaction diagrams nor composite structures explicitly (they use some ad-hoc sequence of actions as well as parallel composition in Object-Z), and while [16] treat composite structures in great detail (also with some operational semantics, which however is not explicated in the paper), they do not consider interactions.

Related approaches formalising connectors such as [4] use a more category-theoretic notation and approach. While this is mathematically more elegant, our approach has the advantage of supporting UML diagrams as they are, without any need of encoding. Moreover, since we support signature morphisms, DOL's modularity constructs (which include constructs for networks of models[3] and their colimit) can be fully used also for UML diagrams.

[3] Technically, a network of (UML) models is a diagram (in the sense of category theory) of logical theories that possibly live in different institutions.

Note that in the sense of [8], our approach is one of asynchronously communicating components. This is because each UML state machine is equipped with its own event pool, which acts as a communication buffer. It should be possible to use the results of [8] for studying (weak) asynchronous compatibility of UML composite structures. This notion ensures that all messages that are sent by some state machine are indeed accepted by another one. Note however that UML has a very simple mechanism to avoid deadlocks of communicating state machines: messages that cannot be processed by a state machine are simply dropped from the event buffer (this is also built-in in our state machine institution). Still, it may be interesting to know whether such situations actually happen or not.

Another important piece of future work is the provision of tool support. The tool Hugo/RT allows checking realisability of interactions for a set of communicating UML state machines, see our previous work [12]. There, a restricted subset of UML interactions is translated into a kind of Büchi automata. While UML composite structures are not considered there, it should be possible to realise our interleaved product construction in Hugo/RT, such that the verification conditions studied in the present paper can be checked as well. The Heterogeneous Tool Set (Hets [15]) provides analysis and proof support for multi-logic specifications in DOL, based on a strong semantic (institution-based) backbone. With Hets, also refinements can be specified and checked. And indeed, Hets can already read in and process XMI files, OMG's interchange format for UML diagrams. Our ultimate goal is to provide a complete integration of a family of UML diagrams into the DOL/Hets ecosystem, and thus to provide tools for UML that are currently not available (see the extensive discussion in [9]): namely comprehensive semantically-grounded support for checking consistency and verification conditions of multi-view UML diagrams.

References

1 Umesh Bellur and V. Vallieswaran. On OO Design Consistency in Iterative Development. In *Proc. 3rd Intl. Conf. Information Technology: New Generations (ITNG'06)*, pages 46–51. IEEE, 2006.

2 Mihai Codescu, Till Mossakowski, Don Sannella, and Andrzej Tarlecki. Specification Refinements: Calculi, Tools, and Applications. *Sci. Comp. Prog.*, 2017. To appear.

3 Gregor Engels, Reiko Heckel, and Jochen Malte Küster. The Consistency Workbench: A Tool for Consistency Management in UML-Based Development. In Perdita Stevens, Jon Whittle, and Grady Booch, editors, *Proc. 6th Intl. Conf. Unified Modeling Language (UML'03)*, volume 2863 of *Lect Notes Comp. Sci.* Springer, 2003.

4 José Luiz Fiadeiro. *Categories for Software Engineering.* Springer, 2005.

5 Martin Glauer. *Institution for Hierarchical UML State Machines.* Master thesis, Otto-von-Guericke-Universität Magdeburg, 2015.

6 Joseph A. Goguen and Rod M. Burstall. Institutions: Abstract Model Theory for Specification and Programming. *J. ACM*, 39:95–146, 1992.

7 Joseph A. Goguen and Grigore Roşu. Institution Morphisms. *Formal Asp. Comp.*, 13:274–307, 2002.

8 Rolf Hennicker, Michel Bidoit, and Thanh-Son Dang. On Synchronous and Asynchronous Compatibility of Communicating Components. In Alberto Lluch-Lafuente and José Proença, editors, *Proc. 18th IFIP WG 6.1 Intl. Conf. Coordination Models and Languages (COORDINATION'16)*, volume 9686 of *Lect. Notes Comp. Sci.*, pages 138–156. Springer, 2016.

9 Alexander Knapp and Till Mossakowski. Multi-view Consistency in UML, 2016. URL: `https://arxiv.org/abs/1610.03960`.

10 Alexander Knapp, Till Mossakowski, and Markus Roggenbach. An Institutional Framework for Heterogeneous Formal Development in UML. A Position Paper. In Rocco De Nicola and Rolf Hennicker, editors, *Software, Services, and Systems. Essays Dedicated to Martin Wirsing on the Occasion of His Retirement from the Chair of Programming and Software Engineering*, volume 8950 of *Lect. Notes Comp. Sci.*, pages 215–230. Springer, 2015.

11 Alexander Knapp, Till Mossakowski, Markus Roggenbach, and Martin Glauer. An Institution for Simple UML State Machines. In Alexander Egyed and Ina Schaefer, editors, *Proc. 18th Intl. Conf. Fundamental Approaches to Software Engineering (FASE'15)*, volume 9033 of *Lect. Notes Comp. Sci.*, pages 3–18. Springer, 2015.

12 Alexander Knapp and Jochen Wuttke. Model Checking of UML 2.0 Interactions. In Thomas Kühne, editor, *Reports Rev. Sel. Papers Ws.s Symp.s MoDELS 2006*, volume 4364 of *Lect. Notes Comp. Sci.*, pages 42–51. Springer, 2007.

13 Kevin Lano, editor. *UML 2 — Semantics and Applications*. Wiley, 2009.

14 Till Mossakowski, Mihai Codescu, Fabian Neuhaus, and Oliver Kutz. The Distributed Ontology, Modelling and Specification Language — DOL. In Arnold Koslow and Arthur Buchsbaum, editors, *The Road to Universal Logic — Festschrift for the 50th Birthday of Jean-Yves Beziau, vol. II*, Studies in Universal Logic. Birkhäuser, 2015.

15 Till Mossakowski, Christian Maeder, and Klaus Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *Proc. 13th Intl. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'07)*, volume 4424 of *Lect. Notes Comp. Sci.*, pages 519–522. Springer, 2007.

16 Iulian Ober and Iulia Dragomir. Unambiguous UML Composite Structures: The OMEGA2 Experience. In Ivana Cerná, Tibor Gyimóthy, Juraj Hromkovic, Keith G. Jeffery, Rastislav Královic, Marko Vukolic, and Stefan Wolf, editors, *Proc. 37th Conf. Current Trends in Theory and Practice of Computer Science (SOFSEM'11)*, volume 6543 of *Lect. Notes Comp. Sci.*, pages 418–430. Springer, 2011.

17 Object Management Group. Unified Modeling Language 2.5. Standard formal/2015-03-01, OMG, 2015. URL: `http://www.omg.org/spec/UML/2.5`.

18 Object Management Group. Distributed Ontology, Modeling, and Specification Language (DOL) 1.0 - Beta. Standard ptc/2016-02-37, OMG, 2016. URL: `http://www.omg.org/spec/DOL/`.

19 Holger Rasch and Heike Wehrheim. Checking Consistency in UML Diagrams: Classes and State Machines. In Elie Najm, Uwe Nestmann, and Perdita Stevens, editors, *Proc. 6th IFIP WG 6.1 Intl. Conf. Formal Methods for Open Object-Based Distributed Systems (FMOODS'03)*, volume 2884 of *Lect. Notes Comp. Sci.*, pages 229–243. Springer, 2003.

20 Tobias Rosenberger. *Relating UML State Machines and Interactions in an Institutional Framework*. Master thesis, Universität Augsburg, 2017.

21 Lutz Schröder, Till Mossakowski, and Christoph Lüth. Type Class Polymorphism in an Institutional Framework. In José Luiz Fiadeiro, editor, *Rev. Sel. Papers 17th Intl. Ws. Recent Trends in Algebraic Development Techniques (WADT'04)*, volume 3423 of *Lect. Notes Comp. Sci.*, pages 234–248. Springer, 2005.

Being Van Kampen in Presheaf Topoi is a Uniqueness Property*

Harald König[1] and Uwe Wolter[2]

1 **Department of Informatics, University of Applied Sciences FHDW Hannover, Freundallee 15, 30173 Hannover, Germany**
harald.koenig@fhdw.de

2 **Department of Informatics, University of Bergen, P.O.Box 7803, 5020 Bergen, Norway**
Uwe.Wolter@uib.no

Abstract

Fibred semantics is the foundation of the model-instance pattern of software engineering. Software models can often be formalized as objects of *presheaf topoi*, i.e, categories of objects that can be represented as algebras as well as coalgebras, e.g., the category of directed graphs. Multimodeling requires to construct *colimits* of models, decomposition is given by *pullback*. Compositionality requires an exact interplay of these operations, i.e., diagrams must enjoy the *Van Kampen* property. However, checking the validity of the Van Kampen property algorithmically based on its definition is often impossible.

In this paper we state a necessary and sufficient yet easily checkable condition for the Van Kampen property to hold in presheaf topoi. It is based on a uniqueness property of path-like structures within the defining congruence classes that make up the colimiting cocone of the models. We thus add to the statement "Being Van Kampen is a Universal Property" by Heindel and Sobociński the fact that the Van Kampen property reveals a set-based structural *uniqueness* feature.

1998 ACM Subject Classification D.3.1. Formal Definitions and Theory, F.3.2. Semantics of Programming Languages

Keywords and phrases Van Kampen Cocone, Presheaf Topos, Fibred Semantics, Mapping Path

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.16

1 Introduction

A presheaf topos is a category, that is based on an algebraic signature with unary operation symbols. Presheaves can also be considered as intersection of algebras and coalgebras [10]. Van Kampen Colimits are a generalization of Van Kampen squares [22]. In [26] we gave a necessary and sufficient condition for a pushout to be a Van Kampen square in a presheaf topos. In the present paper a corresponding criterion is given for all colimiting cocones.

1.1 Motivation

Software engineering and especially model-driven software development requires the decomposition of large models into smaller components, i.e., successful development of large

* This work was partially supported by the Department of Computing, Mathematics, and Physics at Høgskolen i Bergen

applications requires system design fragmentation. Vice versa, a comprehensive viewpoint of a related ensemble of heterogenous software-engineering components is taken up by considering the *amalgamation* (union) of these artefacts modulo their relations amongst each other. This assembly shall not only be carried out on a syntactical level (models), but in the same way on the semantical level (instances). This interplay between assembly and disassembly shows that composition and *correct* decomposition of an instance of a model into instances of the model components always accompany each other. It can be shown that correctness, i.e., *compositionality* [4, 5] is not always guaranteed [20].

Fibred semantics adheres to the model-instance pattern, a standard viewpoint in software engineering: A model M is an object of an appropriate category $\mathbb{C}$, semantics is given by the comma category $\mathbb{C}\downarrow M$. In each object $\tau \in \mathbb{C}\downarrow M$, $\tau : I \to M$, I is the instance structure and τ is its typing. Amalgamation is colimit (of the arrangement of components) and decomposition is performed by taking pullbacks along the cocone morphisms of the colimit.

To wit: Compositionality means that colimit of semantics (instances) is controlled by colimit of syntax (models) such that pullback of the instance colimit retrieves the original instances. Thus compositionality is equivalent to the *Van Kampen property* [7], an abstract characteristic which determines an exactness level for the interaction of colimits and pullbacks. It is thus often necessary to check validity of this property. However, since the definition of the property comes in terms of an equivalence of categories, see Definition 2 in the present paper, algorithmic verification based on the definition is hard even for a finite number of finite models, because the involved comma categories are infinite nevertheless.

Artefacts like UML- or ER-models are based on directed multigraphs, which in turn can be coded as a functor category $Set^{\mathbb{B}}$, where $\mathbb{B}$ has objects E (edges) and V (vertices) and non-identical arrows $s, t : E \to V$. More general metamodels, however, use more sophisticated categories $\mathbb{B}$, such as E-graphs for attributed graphs [3], bipartite graphs for Petri nets [3], or more complex structures for generalized sketches [2]. Hence, $Set^{\mathbb{B}}$ with $\mathbb{B}$ an arbitrary small category, will be the underlying category for the forthcoming investigations.

Constructing colimits in a category $\mathbb{C}$ is an operation on *diagrams*, which are usually coded as functors from a small schema category $\mathbb{I}$ to $\mathbb{C}$. In order to make our results usable for software engineering, we use the older definition for diagrams: Instead of a small category, the schema $\mathbb{I}$ is a finite multigraph and a diagram is a graph morphism from $\mathbb{I}$ to $\mathbb{C}$ [16][1]. The practical construction of colimits relies on *mapping paths*, i.e., chains of pairs of elements that are mapped to each other by the morphisms in the diagram, cf. Definition 3 in Section 3. Thus, colimit computation can easily be carried out algorithmically, if the diagram is finite and consists of finite artefacts.

Summary: While colimit construction is easy, compositionality check (validation of the Van Kampen property) is hard. The *main contribution of the present paper* is a theorem (Theorem 5 in Section 3), which states that a colimit in a presheaf topos has the Van Kampen property if and only if there are no ambiguous mapping paths between any pair of elements of the coproduct of the model artefacts. Thus the implementation of the colimit operation on the model level already provides the material for more efficient compositionality checking.

The paper is organized as follows: Section 2 introduces notation and background information, Section 3 presents the main theorem and applies it to a Software Engineering problem. Section 4 sketches the proof idea: We use a former result, in which a necessary and sufficient criterion is given for pushouts [26]. This result is translated to coequalizers and, finally, lifted to colimits of arbitrary diagrams. Section 5 concludes and discusses future research topics.

More details and elaborated proofs can be found in the underlying technical report [12].

[1] More precisely to the underlying graph of $\mathbb{C}$, see Section 2

1.2 Related Work

The Van Kampen property has its origin in algebraic topology: Topological spaces X can be investigated by a covering family of X which are related by their inclusions. Topological properties are expressed with the help of the fundamental groupoid. The Van Kampen Theorem [18] states that the colimit of the fundamental groupoids of all covering spaces is the fundamental groupoid of X, thus inferring global properties from local ones. The original idea was stated by Seifert [21] for pushouts and was further elaborated by Van Kampen [23].

Inferring global properties from local ones is the heart of sheaf theory [17]. The fibred view on sheaves is discussed in [24]. The application of Van Kampen's ideas to graphical modeling and to Software Engineering was invented in [13, 22] and then further detailed in [3] for the theory of Graph Transformations. That extensive categories and especially topoi are a reasonable playground for these theories is shown in [1, 14].

Amalgamation is a requirement for a collection of artefacts in computer science [4, 5] which has been connected to the Van Kampen property in [26]. The same property is called *exactness* in institution theory [20]. The importance of finding a feasible condition to check the Van Kampen property was caused by investigations of new methods in Graph Transformations [11, 15]. That the Van Kampen property can be characterized as a bicolimit in a comprising span bicategory [7] is a fundamental statement. Moreover, the Van Kampen property has been investigated in more special contexts [9] and can also be described with the help of weak 2-limits in CAT (https://ncatlab.org/nlab/show/van+Kampen+colimit). However, all these characterisations can hardly be applied in practice.

2 Preliminaries

This chapter recapitulates the most important notation for the following elaboration. For any category $\mathbb{C}$, $X \in \mathbb{C}$ means that X is contained in the collection of objects in $\mathbb{C}$. A *diagram* in $\mathbb{C}$ is based on a directed multigraph $\mathbb{I}$, the schema for the diagram. We write $\mathbb{I}_0$ and $\mathbb{I}_1$ for the sets of vertices and edges of $\mathbb{I}$. Formally, a diagram $\mathcal{D} : \mathbb{I} \to \mathcal{U}(\mathbb{C})$ is a graph morphism where $\mathcal{U}$ denotes the forgetful functor assigning to each category its underlying graph. For convenience reasons, however, the forgetful functor will be omitted, i.e., diagrams will be denoted $\mathcal{D} : \mathbb{I} \to \mathbb{C}$. This definition is used instead of the one, where $\mathbb{I}$ is a schema *category* rather than a graph, because it will turn out, that the results in this paper can easier be stated. The notions of (co-)cones and (co-)limits is the same modulo the adjunction $\mathcal{F} \dashv \mathcal{U}$ where $\mathcal{F} : \textsc{Graphs} \to \textsc{Cat}$ assigns to any graph its freely generated category, see [16], III, 4 for more details. Another advantage of this definition occurs in software engineering: Although the schema graph is finite, $\mathcal{F}(\mathbb{I})$ may have infinitely many arrows.

Vertices of $\mathbb{I}$ play the role of indices for diagram objects, hence, we use letters $i, j, \ldots$ for vertices. Edges of $\mathbb{I}$ will be depicted $i \xrightarrow{d} j$ and we write $i = s(d)$, $j = t(d)$ (source and target of d). Images of edges under a diagram $\mathcal{D} : \mathbb{I} \to \mathbb{C}$ will be denoted $\mathcal{D}_i \xrightarrow{\mathcal{D}_d} \mathcal{D}_j$ (slightly deviating from the usual notation $\mathcal{D}(i), \mathcal{D}(d)$, etc).

Let $\mathcal{E}, \mathcal{D} : \mathbb{I} \to \mathbb{C}$ be two diagrams, then a family

$$\tau = (\tau_i : \mathcal{E}_i \to \mathcal{D}_i)_{i \in \mathbb{I}_0}$$

of $\mathbb{C}$-morphisms with $\tau_j \circ \mathcal{E}_d = \mathcal{D}_d \circ \tau_i$ for all edges $i \xrightarrow{d} j$ in $\mathbb{I}_1$ will be called a *natural transformation* between the diagrams and will be denoted in the usual way $\tau : \mathcal{E} \Rightarrow \mathcal{D}$. For any $S \in \mathbb{C}$, $\Delta S : \mathbb{I} \to \mathbb{C}$ denotes the constant diagram, which sends each edge of $\mathbb{I}$ to id_S.

S (as $\mathbb{C}$-object) and ΔS (as diagram) will be used synonymously. Diagrams together with natural transformations constitute the category $\mathbb{C}^{\mathbb{I}}$. Note that $\Delta : \mathbb{C} \to \mathbb{C}^{\mathbb{I}}$ is itself a functor, assigning to each object of $\mathbb{C}$ its constant diagram and to an arrow $f : A \to B$ the "constant" natural transformation $(f)_{i \in \mathbb{I}_0}$.

We assume all categories under consideration to have colimits. The coproduct cocone of a family $(\mathcal{D}_i)_{i \in I}$ of $\mathbb{C}$-objects will be denoted

$$(\mathcal{D}_i \xrightarrow{\subseteq_i} \coprod_{i \in I} \mathcal{D}_i)_{i \in I}.$$

The morphisms $\subseteq_i$ are called coproduct injections. For a family of arrows $(f_i : \mathcal{D}_i \to A)_{i \in I}$ we write $\vec{f} : \coprod_{i \in I} \mathcal{D}_i \to A$ for the resulting unique mediating arrow.

We assume all categories under consideration to have pullbacks. In the sequel, we will work with *chosen pullbacks*, i.e., for each pair of $\mathbb{C}$-arrows $B \xrightarrow{h} A \xleftarrow{k} X$ a choice

$$\begin{array}{ccc} Y & \xrightarrow{h'} & X \\ {\scriptstyle h^*(k)}\downarrow & & \downarrow{\scriptstyle k} \\ B & \xrightarrow[h]{} & A \end{array}$$

of pullback span $(h^*(k), h')$ is determined once and for all. For all $h : B \to A$, $h^*(id_A)$ shall be chosen to be id_B. Whenever we deviate from these choices, this will be emphasized. It is well-known [6] that for fixed $h : B \to A$ chosen pullbacks along h give rise to a (pullback) functor $h^* : \mathbb{C}\downarrow A \to \mathbb{C}\downarrow B$ between comma categories. Pullbacks can be composed, i.e., if $C \xrightarrow{h_2} B \xrightarrow{h_1} A$, then $h_2^* \circ h_1^*$ yields a pullback along $h_1 \circ h_2$, and decomposed, i.e., if $h_1^*(k)$ and $(h_1 \circ h_2)^*(k)$ are computed, the resulting universal arrow from the latter into the former pullback yields a pullback of h_2 and $h_1^*(k)$. Note, that in both cases the automatically appearing pullbacks need not be chosen.

The underlying category for all further considerations is a category of *presheaves*, i.e., the category $\mathbb{G} := Set^{\mathbb{B}}$ (with $\mathbb{B}$ a small (base) category, Set the category of sets and mappings) of covariant functors from $\mathbb{B}$ to Set together with natural transformations between them[2]. We will also use the term "sort" for the objects in $\mathbb{B}$ and the term "operation (symbol)" for the morphisms in $\mathbb{B}$. It is folklore that $\mathbb{G}$ has all colimits and all pullbacks, which are computed sortwise, resp. $\mathbb{G}$ is a topos, i.e. a category with finite limits and colimits, which has exponents and where the subobject functor is representable [6]. $\mathbb{G}$ will thus also be called a *presheaf topos*. E.g., the category of multigraphs is a presheaf topos with $\mathbb{B} = (E \overset{s}{\underset{t}{\rightrightarrows}} V)$ (plus identities). The simplest presheaf topos is Set ($\mathbb{B} = 1$, the one-object-one-morphism category).

In this paper, we will make frequent use of (sortwise) coproducts, i.e., disjoint unions of sets. In order to make argumentations simpler, we will assume that for each $X \in \mathbb{B}$ the artefacts $(\mathcal{D}_i(X))_{i \in \mathbb{I}_0}$ are a priori disjoint, i.e., the coproduct is obtained by simple union.

An important property of presheaf topoi is (infinite) *extensivity*, i.e., the functor

$$\coprod : \prod_{i \in I} \mathbb{G}\downarrow D_i \to \mathbb{G}\downarrow \coprod_{i \in I} D_i \tag{1}$$

[2] Normally presheaves are categories $Set^{\mathbb{B}^{op}}$, i.e., contravariant Set-valued functors. But we prefer the slightly deviating definition, because we found the contravariant version counterintuitive for our work. Clearly, it is easy to switch to the contravariant setting, if one inverts all arrows of $\mathbb{B}$.

assigning to each object $(f_i : A_i \to D_i)_{i\in I}$ in $\prod_{i\in I} \mathbb{G}{\downarrow} D_i$ the object $\coprod_{i\in I} f_i : \coprod_{i\in I} A_i \to \coprod_{i\in I} D_i$ in $\mathbb{G}{\downarrow}\coprod_{i\in I} D_i$, is an equivalence of categories for each index set I and each I-indexed family $(D_i)_{i\in I}$ of objects in $\mathbb{G}$. Its "inverse" arises from constructing pullbacks along coproduct injections. From these facts one derives the stability of coproducts under pullbacks, i.e., if

$$\begin{array}{ccc} A_i & \xrightarrow{a_i} & A \\ {\scriptstyle f_i}\downarrow & & \downarrow{\scriptstyle g} \\ M_i & \xrightarrow{g_i} & M \end{array} \qquad\qquad \begin{array}{ccc} \coprod_{i\in I} A_i & \xrightarrow{\vec{a}} & A \\ {\scriptstyle \coprod_{i\in I} f_i}\downarrow & & \downarrow{\scriptstyle g} \\ \coprod_{i\in I} M_i & \xrightarrow{\vec{g}} & M \end{array} \tag{2}$$

are commutative diagrams, then the squares on the left-hand side are pullbacks for all $i \in I$, if and only if the square on the right-hand side is a pullback [6]. In general topoi, all these statements still hold for finite index sets I (finite extensivity).

3 An Equivalent Condition for the Van Kampen Property

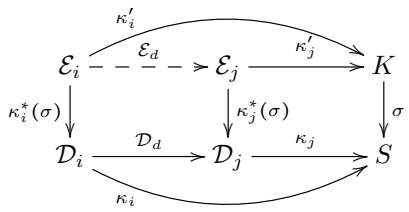
In this chapter we introduce the Van Kampen property and state the main result of this paper, a necessary and sufficient condition for the Van Kampen property to hold in $\mathbb{G} = Set^{\mathbb{B}}$.

3.1 Van Kampen Colimits

A commutative cocone out of a diagram $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ is a natural transformation

$$\kappa : \mathcal{D} \Rightarrow \Delta S. \tag{3}$$

For fixed $i \xrightarrow{d} j$ of $\mathbb{I}_1$, pulling back a $\mathbb{G}$-arrow $K \xrightarrow{\sigma} S$ along κ_i and κ_j yields



(4)

where the right and the outer rectangles are chosen pullbacks, $\mathcal{E}_d$ is the unique completion into the right pullback, and the resulting left square is a pullback by the pullback decomposition property. The left square may, however, not be a chosen one, but it results in diagram $\mathcal{E}$ as well as natural transformation $\kappa^*(\sigma) := (\kappa_i^*(\sigma))_{i\in\mathbb{I}_0} : \mathcal{E} \Rightarrow \mathcal{D}$, whose naturality squares are pullbacks. This fact gives rise to the following definition:

▶ **Definition 1** (Cartesian Transformation)**.** A natural transformation $\tau : \mathcal{E} \Rightarrow \mathcal{D} : \mathbb{I} \to \mathbb{G}$ is called *cartesian* if all naturality squares are pullbacks.

For a fixed diagram $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ let $\mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D}$ be the full subcategory of $\mathbb{G}^{\mathbb{I}} \downarrow \mathcal{D}$ of *cartesian* natural transformations. Thus, by (4), κ^* maps objects of $\mathbb{G} \downarrow S$ to objects in $\mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D}$. Moreover, any arrow $\gamma : \sigma \to \sigma'$ of $\mathbb{G}{\downarrow}S$ yields a family of arrows $(\kappa_i^*(\gamma))$ (universal arrows into pullbacks) of which it can easily be shown that together they yield a cartesian natural transformation $\kappa^*(\gamma) : \kappa^*(\sigma) \to \kappa^*(\sigma')$. Thus κ^* becomes a functor

$$\kappa^* : \mathbb{G}{\downarrow}S \to \mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D}. \tag{5}$$

▶ **Definition 2** (Van Kampen Cocone, [7]). Let $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ be a diagram and $\kappa : \mathcal{D} \Rightarrow \Delta S$ be a commutative cocone. Then κ has the *Van Kampen (VK) Property* ("κ is VK") if functor κ^* is an equivalence of categories.

As usual, a *colimit* (or *colimiting cocone*) is a universal cocone $\kappa : \mathcal{D} \Rightarrow \Delta S$, i.e., for each $T \in \mathbb{G}$ and commutative cocone $\rho : \mathcal{D} \Rightarrow \Delta T$, there is a unique $\mathbb{G}$-morphism $S \xrightarrow{u} T$ such that $\Delta u \circ \kappa = \rho$, i.e., $u \circ \kappa_i = \rho_i$ for all $i \in \mathbb{I}_0$. S is called the *colimit object*.

κ^* has a left-adjoint $\kappa_* : \mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D} \to \mathbb{G}{\downarrow}S$ which assigns to a cartesian natural transformation $\tau : \mathcal{E} \Rightarrow \mathcal{D}$ the unique arrow to S out of the colimit object of the colimiting cocone of $\mathcal{E}$ [22]. I.e., κ_* is the (pseudo-)inverse of κ^*, if the VK property holds. In this case, unit and counit of the adjunction are isomorphisms. Note also that each VK cocone $\mathcal{D} \Rightarrow \Delta S$ is automatically a colimit (apply κ^* to $id_{\Delta S}$ and use the definition of κ_*) such that we can use the terms "Van Kampen cocone" and "Van Kampen colimit" synonymously.

Whereas the counit of this adjunction is always an isomorphism, if pullback functors have right-adjoints (and thus preserve colimits), which is true in every (presheaf) topos [6], the situation is more involved concerning the unit of the adjunction: The easiest example of the VK property arises for the empty diagram. In this case the property translates to the fact, that the *initial object* 0 is strict, i.e., each arrow $A \longrightarrow 0$ is an isomorphism. This is true in all topoi [6]. In the same way, since all presheaf topoi are extensive (cf. Section 2), coproducts have the Van Kampen property. But the unit fails to be an isomorphism for pushouts and coequalizers: Even in *Set* there are easy examples of pushouts which violate the VK property [22]. In *adhesive categories* (and thus in all topoi [14]) pushouts are VK, if one leg is monic, by definition. Vice versa, there are also pushouts with both legs non-monic, which enjoy this property nevertheless [26]. Astonishingly, coequalizers seldom are VK: Consider the shape graph $\mathbf{2} := 1 \overset{d'}{\underset{d}{\rightrightarrows}} 2$ and the diagram $\mathcal{D} : \mathbf{2} \to Set$ with $\mathcal{D}_1 = \{*_1\}, \mathcal{D}_2 = \{*_2\}$. Clearly,

$$\mathcal{D}_1 \rightrightarrows \mathcal{D}_2 \longrightarrow \{*\} \tag{6}$$

is a coequalizer in *Set*. Then the cartesian transformation

$$\tau : (\ \mathcal{E}_1 := \{a, b\} \overset{id}{\underset{k}{\rightrightarrows}} \mathcal{E}_2 := \{a, b\}\ ) \Rightarrow \mathcal{D},$$

with k the non-identical bijection of $\{a, b\}$, is mapped to $id_{\{*\}}$ by κ_*, i.e., $\tau \notin (\kappa^* \circ \kappa_*)(\tau)$.

3.2 Equivalent Condition

As mentioned in the introduction it is important for several software engineering scenarios to find an easily checkable criterion for the Van Kampen property. The presented condition of this paper comes in terms of the mapping behavior of all morphisms $\mathcal{D}_d$ in the diagram.

▶ **Definition 3** (Mapping Path). Let $\mathbb{G} = Set^{\mathbb{B}}$ be a presheaf topos and $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ be a diagram w.r.t. shape graph $\mathbb{I}$. Let $\mathbb{I}_1^{op} := \{d^{op} \mid d \in \mathbb{I}_1\}$.

- A *Path Segment* of sort $X \in \mathbb{B}$ is a triple (y, δ, y') with $\delta \in \mathbb{I}_1 \cup \mathbb{I}_1^{op}$ and[3]

If $\delta = d \in \mathbb{I}_1$	then	$y \in \mathcal{D}_{s(d)}(X), y' = \mathcal{D}_d(y) \in \mathcal{D}_{t(d)}(X)$
If $\delta = d^{op} \in \mathbb{I}_1^{op}$	then	$y' \in \mathcal{D}_{s(d)}(X), y = \mathcal{D}_d(y') \in \mathcal{D}_{t(d)}(X)$

[3] Whenever $i \xrightarrow{d} j \in \mathbb{I}_1$ and we apply a mapping in the family $((\mathcal{D}_d)_X : \mathcal{D}_i(X) \to \mathcal{D}_j(X))_{X \in \mathbb{B}}$, we write $\mathcal{D}_d$ instead of $(\mathcal{D}_d)_X$.

Two path segments (y_1, δ_1, y_1') and (y_2, δ_2, y_2') of sort X are equal if $y_1 = y_2$, $\delta_1 = \delta_2$, and $y_1' = y_2'$. Moreover, two path segments are *weakly equal*, $(y_1, \delta_1, y_1') =_w (y_2, \delta_2, y_2')$ in symbols, if $(y_1, \delta_1, y_1') = (y_2, \delta_2, y_2')$ or $(y_1, \delta_1, y_1') = (y_2', \delta_2^{op}, y_2)$.[4]

- A *Non-empty Mapping Path in* $\mathcal{D}$ of sort $X \in \mathbb{B}$ is a sequence

$$P = [(y_0, \delta_0, y_1), (y_1, \delta_1, y_2), (y_2, \delta_2, y_3), \ldots, (y_{n-1}, \delta_{n-1}, y_n)]$$

of path segments of sort X, where any third component of a segment coincides with the first component of its successor segment[5], and where $n \geq 1$. We say that the above path connects y_0 with y_n in $\mathcal{D}$.
- For each $y \in \mathcal{D}_i(X)$, where $i \in \mathbb{I}_0$ and $X \in \mathbb{B}$, we say that the *Empty Mapping Path* [ ] of sort X connects y with itself in $\mathcal{D}$.
- Two paths are equal, if they have the same length and are segmentwise equal.
- A mapping path is *proper* if there are no two distinct path segments that are weakly equal.

Examples of mapping paths for graphs are depicted in Figure 1 (the complete meaning of the contents of Figure 1 will be explained in the next section): There are two paths (one along the dashed path segments, the other one along the dotted segments) both connecting vertex "Sort" with vertex "Type". Each arrow depicts a path segment with first component the arrow's source and third component its target. The middle component is annotated near the arrows, resp., their names will be explained in the next section, as well.

For any $X \in \mathbb{B}$, any $i, j \in \mathbb{I}_0$ and any $z \in \mathcal{D}_i(X)$, $z' \in \mathcal{D}_j(X)$ we write $z \equiv_X z'$ ($z \equiv_X^p z'$), if there is a mapping path (proper mapping path) of sort X connecting z with z'. It is easy to see that $\equiv = \equiv^p$ and that this relation is a congruence relation on $\coprod_{i \in \mathbb{I}_0} \mathcal{D}_i$ (i.e., a family of equivalence relations $(\equiv_X)_{X \in \mathbb{B}}$ compatible with operations of $\mathbb{B}$), because paths can be concatenated and reversed. Moreover, it is well-known [16] that the colimiting cocone of diagram $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ is given by

$$\mathcal{D} \overset{\kappa}{\Rightarrow} (\coprod_{i \in \mathbb{I}_0} \mathcal{D}_i)/\equiv \; = (\coprod_{i \in \mathbb{I}_0} \mathcal{D}_i)/\equiv^p \tag{7}$$

where $\kappa_i = [\ ]_\equiv \circ \subseteq_i$ with $[\ ]_\equiv$ the canonical morphism. In the present paper we will show that mapping paths also play a crucial role for a simpler characterization of the Van Kampen property. The following examples hint at this connection.

▶ **Example 4.** Let $\mathbb{G} = Set$.

1. In (6) there are proper mapping paths [] and $[(*_2, d^{op}, *_1), (*_1, d', *_2)]$ both connecting $*_2$ with itself.
2. The shape graph $1 \xleftarrow{d} 0 \xrightarrow{d'} 2$ yields pushouts. The easiest example of a non-VK pushout arises from $\mathcal{D}_0 = \{x, y\}, \mathcal{D}_1 = \{*_1\}, \mathcal{D}_2 = \{*_2\}$, cf. [22]. In this case, we obtain two different proper mapping paths $[(*_1, d^{op}, x), (x, d', *_2)]$ and $[(*_1, d^{op}, y), (y, d', *_2)]$ both connecting $*_1$ and $*_2$ in $\mathcal{D}$.
3. Let $\mathbb{I} =$  consist of one vertex and one loop. I.e., diagrams depict endomorphisms $f : A \to A$. It is astonishing that even the colimiting cocone $\mathcal{D} \Longrightarrow \{*\}$ with $\mathcal{D}_d = id_{\{*\}}$ is not VK: Take $\mathcal{E} = (\ \{a, b\} \xrightarrow{k} \{a, b\}\ )$ (with k the non-identity bijection of $\{a, b\}$),

[4] $(d^{op})^{op} := d$.

[5] By the introductory remarks on disjointness of artefacts, this means that the third component and the successor's first component are elements of the same $\mathcal{D}_i$.

$\tau : \{a, b\} \to \{*\}$, then $\mathcal{E}$'s colimit is a singleton. In this example, we have two proper mapping paths [] and $[(*, d, *)]$ in $\mathcal{D}$ both connecting $*$ with itself. Note that this is just another presentation of example (6), since the colimit of $f : A \to A$ can be obtained by the coequalizer of $A \overset{id_A}{\underset{f}{\rightrightarrows}} A$).

4. $\mathcal{D} = (\ \{x\} \overset{g}{\underset{f}{\rightrightarrows}} \{y, z\}\ )$ with $f(x) = y, g(x) = z$ has the VK property, which can be checked by elementary means based on Definition 2. There is exactly one proper mapping path connecting y and z, namely $(y, f^{op}, x), (x, g, z)$. Moreover, there is exactly one proper path connecting y with itself (namely the empty one, the hypothetical path $(y, f^{op}, x), (x, f, y)$ is not proper, see Definition 3). In the same way x has only one path back to itself, namely the empty one (the hypothetical path $(x, f, y), (y, f^{op}, x)$ is not proper).

As suggested by these examples, *uniqueness of proper mapping paths* between two elements of the same sort X in the sets $(\mathcal{D}_i(X))_{i \in \mathbb{I}_0}$ is a crucial feature for the Van Kampen property to hold. Indeed, we will prove

▶ **Theorem 5** (Characterization of VK Cocones as Uniqueness Property). *Let $\mathbb{G} = Set^{\mathbb{B}}$ be a presheaf topos and $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ be a diagram. Let $\mathcal{D} \overset{\kappa}{\Rightarrow} \Delta S$ be a colimiting cocone. The cocone is a Van Kampen cocone if and only if for all $X \in \mathbb{B}$, all $i, j \in \mathbb{I}_0$ and all $z \in \mathcal{D}_i(X)$, $z' \in \mathcal{D}_j(X)$: There are no two different proper mapping paths in $\mathcal{D}$ connecting z and z'.*

Since, in colimit computations, all mapping paths need to be computed (see (7)), and – according to Theorem 5 – the Van Kampen property can be checked by means of mapping paths, algorithmic verification of the Van Kampen property can be carried out in the background of colimit computation. In the technical report [12], we further simplify the condition of Theorem 5 and thus simplify the algorithm: We state conditions on the morphisms of $\mathcal{D}$, under which the Van Kampen property always holds (e.g., if all $\mathcal{D}_d$ are monomorphisms and enjoy a certain kind of image-disjointness) and we identify special shapes of schema graph $\mathbb{I}$, where a significantly smaller subset of indices i has to be tested for path uniqueness.

3.3 Application of Theorem 5

In order to demonstrate the benefits of this criterion, we consider a more substantial example than the ones in Example 4. We let $\mathbb{B} = (\ E \overset{s}{\underset{t}{\rightrightarrows}} V\ )$ (id_E and id_V not shown), thus our base presheaf topos is $\mathbb{G} = Set^{\mathbb{B}}$, the category of directed multigraphs. In the sequel, we depict vertices as rectangles and edges are arrows pointing from its source to its target. In Figure 1 the three highlighted graphs[6] $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$ depict meta-models for type systems:

- $\mathcal{D}_1$ represents parts of the domain of *algebraic specifications*: Operations have an arbitrary number of sort-typed input parameters and exactly one return parameter.
- In $\mathcal{D}_2$ terminology of *abstract data types* is used: Functions have an arbitrary number of typed input and return parameters, resp.
- $\mathcal{D}_3$ is the object-oriented view: Interfaces own operations, which have inputs and one return parameter typed in interfaces, resp. Methods implement operations, their input parameters may be of specialized type.

[6] These are not just graphs since they contain "multiplicity constraints". They can be formalized, actually, as generalized sketches, i.e., graphs with diagrammatic predicates, in the sense of [2].

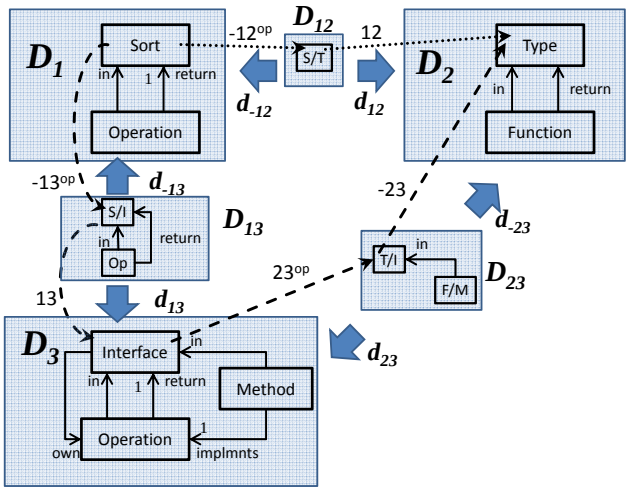
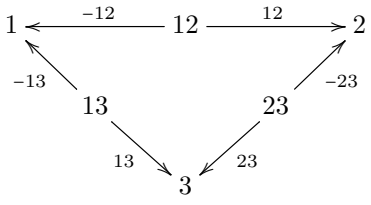


Figure 1 A diagram of metamodels and two mapping paths.

Figure 1 represents a *multimodeling* scenario [19]. Reasoning about these collective models (the multimodel) as one artefact requires matching of different terminology of each of the model graphs: Sameness of terminology in graphs $\mathcal{D}_1$ and $\mathcal{D}_2$ is formally enabled by defining a relation on $\mathcal{D}_1 \times \mathcal{D}_2$ by means of auxiliary graph $\mathcal{D}_{12}$, which consists of exactly one vertex S/T, $d_{-12}(S/T) = Sort$, $d_{12}(S/T) = Type$, such that span $\mathcal{D}_1 \overset{d_{-12}}{\leftarrow} \mathcal{D}_{12} \overset{d_{12}}{\rightarrow} \mathcal{D}_2$ specifies sameness of terms "Sort" and "Type" in graphs $\mathcal{D}_1$, $\mathcal{D}_2$ and no other commonalities. In the same way span $\mathcal{D}_1 \overset{d_{-13}}{\leftarrow} \mathcal{D}_{13} \overset{d_{13}}{\rightarrow} \mathcal{D}_3$ specifies sameness of terms "Sort" and "Interface" (in $\mathcal{D}_1$ and $\mathcal{D}_3$) as well as "Operation" (in both graphs) together with the in- and return-relationships. Moreover, relation "in" of term "Method" in $\mathcal{D}_3$ is declared to be equal to property "in" of term "Function" in $\mathcal{D}_2$ via span $\mathcal{D}_3 \overset{d_{23}}{\leftarrow} \mathcal{D}_{23} \overset{d_{-23}}{\rightarrow} \mathcal{D}_2$.

We now describe a scenario, in which colimit computation of the graphs in Figure 1 and amalgamation of instances typed over these graphs is important. It is common to reason about the multimodel by imposing constraints that spread over different models. We could, e.g., claim that "The return type of a method's implemented operation (as specified in $\mathcal{D}_3$) has to be contained in the list of return types of the corresponding function (as specified in $\mathcal{D}_2$)". In order to check this inter-model constraint, it is necessary to construct the diagram's colimit. Formally, for schema graph $\mathbb{I}$ =



we obtain diagram $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ and construct the colimiting cocone $\mathcal{D} \overset{\kappa}{\Rightarrow} \Delta S$.

Assume now that we want to check consistency of given typed instances $\tau_i : \mathcal{E}_i \to \mathcal{D}_i$, $i \in \{1, 2, 3\}$ against the above formulated constraint. For this we have to declare sameness of elements within $\mathcal{E}_1, \mathcal{E}_2$, and $\mathcal{E}_3$ with the help of new relating typing morphisms $\tau_k : \mathcal{E}_k \to \mathcal{D}_k$,

$k \in \{12, 13, 23\}$ and spans $\mathcal{E}_1 \overset{e_{-12}}{\leftarrow} \mathcal{E}_{12} \overset{e_{12}}{\rightarrow} \mathcal{E}_2$, $\mathcal{E}_1 \overset{e_{-13}}{\leftarrow} \mathcal{E}_{13} \overset{e_{13}}{\rightarrow} \mathcal{E}_3$, and $\mathcal{E}_2 \overset{e_{-23}}{\leftarrow} \mathcal{E}_{23} \overset{e_{23}}{\rightarrow} \mathcal{E}_3$. Of course, all τ_k have to be compatible with model matching and sameness declaration within $\mathcal{E}_1, \mathcal{E}_2$, and $\mathcal{E}_3$, i.e., we obtain a natural transformation $\tau : \mathcal{E} \Rightarrow \mathcal{D}$ between diagrams of type $\mathbb{I} \to \mathbb{G}$. Consistency checking is then carried out by constructing the colimit object K of $\mathcal{E}$ and checking whether the resulting typing arrow $\sigma : K \to S$ fulfills the constraint, see [19].

Let us momentarily ignore constraint checking and just consider the relation between this amalgamated instance σ and the original component instances $(\tau_i)_{i \in \{1,2,3,12,13,23\}}$: It is necessary to faithfully recover all τ_i from σ, otherwise we would loose information about the origin of the elements in the domain of σ. This means that we require, for all i, $\kappa_i^*(\sigma) \cong \tau_i$, i.e., the Van Kampen property for the cocone κ has to hold. However, it turns out, that the property is violated: This can be seen by considering the following instance constellation (we write $x{:}T$ whenever $\tau\_(x) = T$): Let $\mathcal{E}_1(V) = \{s{:}Sort, s'{:}Sort\}, \mathcal{E}_1(E) = \varnothing$, $\mathcal{E}_2(V) = \{t_1{:}Type, t_2{:}Type\}, \mathcal{E}_2(E) = \varnothing$, and $\mathcal{E}_3(V) = \{\bar{i}{:}Interface, i{:}Interface\}, \mathcal{E}_3(E) = \varnothing$. One may now declare sameness of elements within $\mathcal{E}_1, \mathcal{E}_2$, and $\mathcal{E}_3$ as follows

$$s = t_1, s' = t_2 \text{ by span } (e_{-12}, e_{12}); s = i, s' = \bar{i} \text{ by } (e_{-13}, e_{13}); t_1 = \bar{i}, t_2 = i \text{ by } (e_{-23}, e_{23}).$$

This is established as described above, e.g., graph $\mathcal{E}_{12}$ consists of two vertices $1{:}S/T$ and $2{:}S/T$. Graph morphisms e_{-12} maps $1{:}S/T \mapsto s$ and $2{:}S/T \mapsto s'$ wheras e_{12} maps $1{:}S/T \mapsto t_1$ and $2{:}S/T \mapsto t_2$. We omit the obvious formal definitions of the other two spans.

Unfortunately, by transitivity, this matching also yields $s = s'$, an unwanted anomaly. But in practice this effect may happen, if two modelers work separately: One modeler might define matches (e_{-12}, e_{12}) and (e_{-13}, e_{13}) and, independently and inadvertently, the second modeler defines the match (e_{-23}, e_{23}). The inconsistent matching yields a colimit K of $\mathcal{E}$ with one vertex only, because each sort/type/interface is connected with each other along mapping paths. Clearly, $\kappa_i^*(\sigma) \not\cong \tau_i$ since κ_i are monomorphisms, hence the domains of $\kappa_i^*(\sigma)$ are singleton sets, as well.

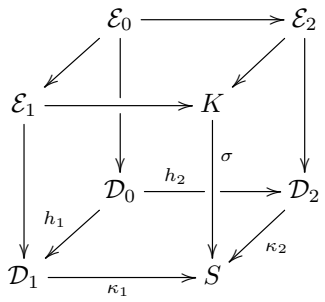
In this simple example, a small instance constellation allowed for the detection of a VK violation witness. However, it is hard to determine such witnesses in more complex examples. In these cases, Theorem 5 is a more reliable indicator for VK validity or violation, because we do not need to find violating instance constellations. Instead, the violation of the VK property can be detected by analysing mapping path structures of the metamodels only. In the present example, the indicator are the *two different proper mapping paths* of sort V shown in Figure 1 both connecting "Sort" and "Type" (one is depicted by dashed, the other one by dotted arrows) such that Theorem 5 immediately yields violation of the Van Kampen property. At least from this example we derive the slogan that *the Van Kampen property holds, if there is no redundant matching information* in $\mathcal{D}$. It is easy to see that the negative effect vanishes if we reduce the diagram accordingly, i.e. if we erase matching via $\mathcal{D}_{12}$ since this information is already contained in the transitive closure of matchings $\mathcal{D}_{13}$ and $\mathcal{D}_{23}$. In this way, the above mentioned modelers can indeed work independently!

4 An Outline of the Proof of Theorem 5

In this section, we sketch the main steps for the proof of our main theorem. Each step is given by a Lemma for which detailed proofs can be found in the technical report [12].

4.1 Pushouts

Often, the Van Kampen property for pushouts is formulated as follows: A pushout of a diagram $\mathcal{D}_1 \xleftarrow{h_1} \mathcal{D}_0 \xrightarrow{h_2} \mathcal{D}_2$ is said to have the Van Kampen property if for any commutative cube



with this pushout in the bottom and back faces pullbacks, the front faces are pullbacks if and only if the top face is a pushout. In [26] we already stated a characterization of the Van Kampen property for pushouts based on this definition. It comes in terms of cyclic mapping structures within $\mathcal{D}_0$:

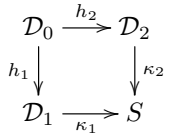
▶ **Definition 6** (Domain Cycle, [15]). Consider a span $\mathcal{D}_1 \xleftarrow{h_1} \mathcal{D}_0 \xrightarrow{h_2} \mathcal{D}_2$ in $\mathbb{G} = Set^{\mathbb{B}}$. For $X \in \mathbb{B}$ we call a sequence $[x_0, x_1, \ldots, x_{2k+1}]$ of elements of $\mathcal{D}_0(X)$ a *domain cycle* (for the span (h_1, h_2)) (of sort X), if $k \in \mathbb{N}$ and the following conditions hold:

1. $\forall j \in \{0, 1, \ldots, 2k+1\} : x_j \neq x_{j+1}$
2. $\forall i \in \{0, \ldots, k\} : h_1(x_{2i}) = h_1(x_{2i+1})$
3. $\forall i \in \{0, \ldots, k\} : h_2(x_{2i+1}) = h_2(x_{2i+2})$

where $2k+2 := 0$. A domain cycle is *proper* if $x_i \neq x_j$ for all $0 \leq i < j \leq 2k+1$.

The main outcome of [26] is the following fact:

▶ **Lemma 7** (Condition for VK Pushouts). *A pushout*



in $\mathbb{G} = Set^{\mathbb{B}}$ is a Van Kampen cocone iff there is no proper domain cycle for (h_1, h_2). ◀

This result can be proven by means of elementary set-based arguments [15], but also by investigating forgetful functors between categories of descent data [8] for general topoi [26].

It is easy to see that the above definition for pushouts is an instance of the general definition of Van Kampen colimits in Definition 2:

- If the front faces are pullbacks, then the back faces are the result of applying κ^*. Then the counit $\varepsilon : \kappa_* \circ \kappa^* \Rightarrow Id$ of adjunction is an isomorphism if and only if the cube's top face is already a pushout.
- If the top face is a pushout, then (up to isomorphism) σ is the result of applying κ_*. Then the unit $\eta : Id \Rightarrow \kappa^* \circ \kappa_*$ is an isomorphism if and only if $\kappa^*(\sigma)$ produces the original cube up to isomorphism, i.e., the original front faces are pullbacks.

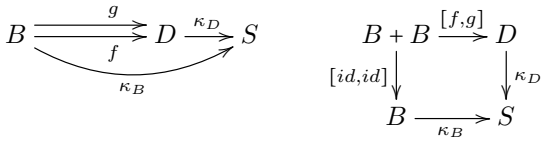
Hence, the two implications "Front face pullbacks iff top face pushout" actually reflect the two statements "The counit is an isomorphism" and "The unit is an isomorphism".

Hence, Lemma 7 is a good starting point for the proof of Theorem 5: We transfer the knowledge to special mapping paths in coequalizer diagrams (Section 4.2) and from there to mapping paths in arbitrary colimits (Section 4.3).

4.2 From Pushouts to Coequalizers

The transfer from pushouts to coequalizers is accomplished in two steps. The first step connects the VK property for coequalizers and pushouts:

▶ **Lemma 8.** *Let $\mathbb{G}$ be a general topos and $B \overset{g}{\underset{f}{\rightrightarrows}} D$ be two arrows in $\mathbb{G}$. Let two arrows $\kappa_D : D \to S$ and $\kappa_B : B \to S$ be given such that the diagrams*



are commutative, resp.

1. *The left diagram is a coequalizer if and only if the right diagram is a pushout.*
2. *The left diagram is a VK cocone if and only if the right diagram is.*

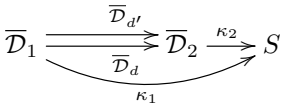
Proof. (1) is well-known [16]. (2) is proven by means of Definition 2, where the transfer is possible, because topoi are (finitely) extensive, cf. Section 2, and especially because of property (2). ◀

The second step establishes a connection between domain cycles and mapping paths. It comes in terms of *disjoint* mapping paths, i.e., paths P_1 and P_2 for which non of the path segments in P_1 is weakly equal[7] to a path segment in P_2. The proof is rather technical and will be omitted, see [12], Lemma 14.

▶ **Lemma 9** (Domain Cycles vs. Mapping Paths). *Let $\mathbb{G} = Set^{\mathbb{B}}$, f and g as in Lemma 8, and $X \in \mathbb{B}$. There is a proper domain cycle of sort X for $B \overset{[id,id]}{\longleftarrow} B + B \overset{[f,g]}{\longrightarrow} D$, if and only if there are $z, z' \in D(X)$ and two disjoint proper mapping paths connecting z and z'.* ◀

Lemmas 7, 8, and 9 yield

▶ **Corollary 10** (Condition for VK Coequalizers). *Let $\mathbb{G} = Set^{\mathbb{B}}$, let $\mathbf{2}$ be the schema graph $1 \overset{d'}{\underset{d}{\rightrightarrows}} 2$, and $\overline{\mathcal{D}} : \mathbf{2} \to \mathbb{G}$. The coequalizer diagram*



has the Van Kampen property, if and only if for all $X \in \mathbb{B}$ and all $z, z' \in \overline{\mathcal{D}}_2(X)$: There are no two disjoint proper mapping paths of sort X in $\overline{\mathcal{D}}$ connecting z and z'. ◀

Recall the already made observations in Example 4, 1. and 4., which confirm this statement.

[7] Recall the definition of weak equality in Definition 3.

4.3 From Coequalizers to Colimits

It is well-known [16], that the colimit of $\mathcal{D} : \mathbb{I} \to \mathbb{G}$ can be computed by constructing the coequalizer of

$$\coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} \underset{\vec{\mathcal{D}}_d}{\overset{\vec{id}}{\rightrightarrows}} \coprod_{j \in \mathbb{I}_0} \mathcal{D}_j, \tag{8}$$

where $\vec{\mathcal{D}}_d$ and $\vec{id}$ are mediators out of the involved coproducts:

$$\begin{array}{ccc} \coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} & \xrightarrow{\vec{\mathcal{D}}_d} & \coprod_{j \in \mathbb{I}_0} \mathcal{D}_j \\ {\scriptstyle \subseteq_{i,d}}\uparrow & & \uparrow{\scriptstyle \subseteq_j} \\ \mathcal{D}_i & \xrightarrow{\mathcal{D}_d} & \mathcal{D}_j \end{array} \qquad \begin{array}{ccc} \coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} & \xrightarrow{\vec{id}} & \coprod_{i \in \mathbb{I}_0} \mathcal{D}_i \\ {\scriptstyle \subseteq_{i,d}}\uparrow & & \uparrow{\scriptstyle \subseteq_i} \\ \mathcal{D}_i & = & \mathcal{D}_i \end{array} \tag{9}$$

(for all edges $i \xrightarrow{d} j$ in $\mathbb{I}_1$).[8]

Let $\overline{\mathcal{D}} : \mathbf{2} \to \mathbb{G}$ be the functor mapping $\mathbf{2}$ to the objects and arrows in (8), where schema graph $\mathbf{2}$ is given as before (cf. e.g. Corollary 10). Then a technical analysis shows that we can combine mapping paths of $\mathcal{D}$ with special mapping paths of $\overline{\mathcal{D}}$ (again, we omit the proof and refer to [12]):

▶ **Lemma 11.** *Let $\mathbb{G} = Set^{\mathbb{B}}$ and $X \in \mathbb{B}$. The following statements are equivalent:*

- *$\forall i, j \in \mathbb{I}_0 : \forall z \in \mathcal{D}_i(X), \forall z' \in \mathcal{D}_j(X)$: There are no two disjoint proper mapping paths in $\mathcal{D}$ connecting z and z'.*
- *$\forall z, z' \in \overline{\mathcal{D}}_2(X) = \coprod_{j \in \mathbb{I}_0} \mathcal{D}_j(X)$: There are no two disjoint proper mapping paths in $\overline{\mathcal{D}}$ connecting z and z'.* ◀

The main part of the proof of Theorem 5 is to carry over the VK property for the coequalizer of (8) to its underlying colimiting diagram for $\mathcal{D}$.

▶ **Lemma 12.** *For $\mathbb{G} := Set^{\mathbb{B}}$, the cocone (3) is VK if and only if the cocone*

$$\coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} \underset{\vec{\mathcal{D}}_d}{\overset{\vec{id}}{\rightrightarrows}} \coprod_{j \in \mathbb{I}_0} \mathcal{D}_j \xrightarrow{\overline{\kappa}} S \qquad \overline{\kappa}' : \coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} \to S \tag{10}$$

resulting from constructing the coequalizer in (8) is VK.

Proof: Let $\kappa^* : \mathbb{G} \downarrow S \to \mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D}$ be the functor introduced in (5) and $\overline{\kappa}^* : \mathbb{G} \downarrow S \to \mathbb{G}^{\mathbf{2}} \Downarrow \overline{\mathcal{D}}$ be the corresponding functor for the colimiting cocone in (10). Using (1) and (2), one can show that for each cartesian $\tau : \mathcal{E} \Rightarrow \mathcal{D}$ the squares

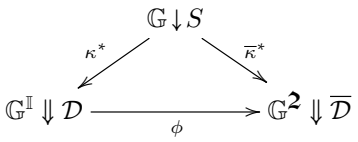
$$\begin{array}{ccc} \coprod_{d \in \mathbb{I}_1} \mathcal{E}_{s(d)} & \xrightarrow{\vec{id}} & \coprod_{i \in \mathbb{I}_0} \mathcal{E}_i \\ {\scriptstyle \coprod_{d \in \mathbb{I}_1} \tau_{s(d)}}\downarrow & & \downarrow{\scriptstyle \coprod_{i \in \mathbb{I}_0} \tau_i} \\ \coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} & \xrightarrow{\vec{id}} & \coprod_{i \in \mathbb{I}_0} \mathcal{D}_i \end{array} \qquad \begin{array}{ccc} \coprod_{d \in \mathbb{I}_1} \mathcal{E}_{s(d)} & \xrightarrow{\vec{\mathcal{E}}_d} & \coprod_{j \in \mathbb{I}_0} \mathcal{E}_j \\ {\scriptstyle \coprod_{d \in \mathbb{I}_1} \tau_{s(d)}}\downarrow & & \downarrow{\scriptstyle \coprod_{j \in \mathbb{I}_0} \tau_j} \\ \coprod_{d \in \mathbb{I}_1} \mathcal{D}_{s(d)} & \xrightarrow{\vec{\mathcal{D}}_d} & \coprod_{j \in \mathbb{I}_0} \mathcal{D}_j \end{array}$$

are pullbacks, i.e., there is the assignment $\tau \mapsto (\coprod_{d \in \mathbb{I}_1} \tau_{s(d)}, \coprod_{i \in \mathbb{I}_0} \tau_i)$. It can be shown with elementary arguments that it extends to an equivalence of categories:

$$\phi : \mathbb{G}^{\mathbb{I}} \Downarrow \mathcal{D} \cong \mathbb{G}^{\mathbf{2}} \Downarrow \overline{\mathcal{D}}.$$

[8] Note that in the left coproduct of (8) an object $\mathcal{D}_i$ occurs as often as there are edges d leaving i in $\mathbb{I}$. Moreover, $\subseteq_{i,d}$ in (9) denotes the embedding of $\mathcal{D}_i$ into its appropriate copy, namely the source of $\mathcal{D}_d$.

Moreover, the colimit construction principle, see (8), yields commutativity of



up to natural isomorphism, hence, by Definition 2, the desired result. ◀

4.4 Combining the Results

We are now ready to prove Theorem 5 for disjoint proper mapping paths. This follows by combining Lemma 12, Corollary 10 and Lemma 11. Afterwards we can get rid of disjointness by showing that any two proper mapping paths connecting the same two elements also admit two disjoint proper paths (probably connecting two different elements). ◀

5 Conclusion and Future Work

In general, arbitrary diagrams in arbitrary categories are not VK. Even if we restrict to presheaf topoi, many diagrams are not VK. In the paper we presented a feasible condition (Theorem 5) to check if a diagram in a presheaf topos is VK or not.

As suggested by the example in Section 3.3, modelers may well work with a non-VK-diagram (of software models), if they have a common understanding of the used natural transformation $\tau : \mathcal{E} \Rightarrow \mathcal{D}$, i.e., if they know how to avoid "twisting anomalies" as shown in the example. Hence, the natural next step will be to look for feasible conditions that a given $\tau : \mathcal{E} \Rightarrow \mathcal{D}$ is in the image of κ^*, even if the diagram is not VK. We may allow non-uniqueness of mapping paths in diagrams of models, but then paths in the diagram of instances have to be exact copies of them, i.e., path liftings from models to instances must behave like discrete fibrations. It is worth to underline that the instances we get from a given "indexed semantics" via a corresponding variant of the Grothendieck construction [25] are always contained in the image of κ^* up to isomorphism.

The ultimate goal, however, is to find a categorical counterpart for the different paths criterion (Theorem 5), which states a necessary and sufficient condition for the Van Kampen property in more general categories. Is such a condition significantly different from the bilimit condition mentioned in the introduction and the universal property in [7]?

References

1 M. Bunge and S. Lack. Van Kampen Theorems for Topoi. *Advances in Mathematics*, 179:291 – 317, 2003.

2 Z. Diskin and U. Wolter. A Diagrammatic Logic for Object-Oriented Visual Modeling. *Electr. Notes Theor. Comput. Sci.*, 203(6):19–41, 2008. doi:10.1016/j.entcs.2008.10.041.

3 H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformations*. Springer, 2006.

4 Hartmut Ehrig, M. Grosse-Rhode, and U. Wolter. Applications of Category Theory to the Area of Algebraic Specification in Computer Science. *Applied Categorical Structures*, 6:1–35, 1998.

5 Jose Luiz Fiadeiro. *Categories for Software Engineering*. Springer, 2005.

6 Robert Goldblatt. *Topoi: The Categorial Analysis of Logic*. Dover Publications, 1984.

7 T. Heindel and P. Sobociński. Van Kampen Colimits as Bicolimits in Span. In A. Kurz, M. Lenisa, and A. Tarlecki, editors, *Algebra and Coalgebra in Computer Science*, volume 5728 of *Lecture Notes in Comput. Sci.*, pages 335–349. Springer Berlin / Heidelberg, 2009. doi:10.1007/978-3-642-03741-2_23.

8 G Janelidze and W. Tholen. Facets of Descent, I. *Appl. Categorical Structures*, 2:245–281, 1994. doi:10.1007/BF00878100.

9 Wolfram Kahl. Collagories: Relation-algebraic Reasoning for Gluing Constructions. *J. Log. Algebr. Program.*, 80(6):297–338, 2011. doi:10.1016/j.jlap.2011.04.006.

10 Wolfram Kahl. *Categories of Coalgebras with Monadic Homomorphisms*, pages 151–167. Springer, Berlin, Heidelberg, 2014. doi:10.1007/978-3-662-44124-4_9.

11 Harald König, Michael Löwe, Christoph Schulz, and Uwe Wolter. Van Kampen Squares for Graph Transformation. In *Graph Transformation - 7th International Conference, ICGT 2014, Held as Part of STAF 2014, York, UK, July 22-24, 2014. Proceedings*, pages 222–236, 2014. doi:10.1007/978-3-319-09108-2_15.

12 Harald König and U. Wolter. Van Kampen Colimits in Presheaf Topoi. Technical report, University of Applied Sciences, FHDW Hannover, 2016. URL: http://fhdwdev.ha.bib.de/public/papers/02016-02.pdf.

13 S. Lack and P. Sobociński. Adhesive Categories. In *Foundations of Software Science and Computation Structures (FoSSaCS '04)*, volume 2987, pages 273–288. Springer, 2004. doi:10.1007/978-3-540-24727-2_20.

14 S. Lack and P. Sobociński. Toposes are Adhesive. *Lecture Notes in Comput. Sci.*, 4178:184–198, 2006. doi:10.1007/11841883_14.

15 Michael Löwe. Van Kampen Pushouts for Sets and Graphs. Technical report, University of Applied Sciences, FHDW Hannover, 2010.

16 Saunders Mac Lane. *Categories for the Working Mathematician, Second edition.* Springer, 1998.

17 Saunders MacLane and Ieke Moerdijk. *Sheaves in Geometry and Logic. A first introduction to topos theory.* Springer, 1992.

18 J.P. May. *A Concise Course in Algebraic Topology.* Chicago Lectures in Mathematics. The University of Chicago Press, 1999. doi:10.1007/978-3-642-17336-3.

19 Mehrdad Sabetzadeh, Shiva Nejati, Sotirios Liaskos, Steve M. Easterbrook, and Marsha Chechik. Consistency Checking of Conceptual Models via Model Merging. In *Requirements Engineering Conference*, pages 221–230, 2007.

20 Donald Sannella and Andrzej Tarlecki. *Foundations of Algebraic Specification and Formal Software Development.* Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2012. doi:10.1007/978-3-642-17336-3.

21 Herbert Seifert. Konstruktion dreidimensionaler geschlossener Räume. *Dissertation, University of Dresden*, 1931.

22 P. Soboczińsky. Deriving Process Congruences from Reaction Rules. Technical Report DS-04-6, BRICS Dissertation Series, 2004.

23 E. R. van Kampen. On the Connection between the Fundamental Groups of some Related Spaces. *American Journal of Mathematics*, 55:261 – 267, 1933.

24 A. Vistoli. Grothendieck Topologies, Fibered Categories and Descent Theory. *Fundamental Algebraic Geometry, Math. Surveys Monogr., Amer. Math. Soc., Providence, RI, 2005*, 123:1 – 104, 2005.

25 U. Wolter and Z. Diskin. From Indexed to Fibred Semantics – The Generalized Sketch File –. Reports in Informatics 361, Dep. of Informatics, University of Bergen, 2007.

26 U. Wolter and H. König. Fibred Amalgamation, Descent Data, and Van Kampen Squares in Topoi. *Applied Categorical Structures*, 23(3):447 – 486, 2015. doi:10.1007/s10485-013-9339-2.

Custom Hypergraph Categories via Generalized Relations*

Dan Marsden[1] and Fabrizio Genovese[2]

1 **Department of Computer Science, University of Oxford, UK**
2 **Department of Computer Science, University of Oxford, UK**

Abstract

Process theories combine a graphical language for compositional reasoning with an underlying categorical semantics. They have been successfully applied to fields such as quantum computation, natural language processing, linear dynamical systems and network theory. When investigating a new application, the question arises of how to identify a suitable process theoretic model.

We present a conceptually motivated parameterized framework for the construction of models for process theories. Our framework generalizes the notion of binary relation along four axes of variation, the truth values, a choice of algebraic structure, the ambient mathematical universe and the choice of proof relevance or provability. The resulting categories are preorder-enriched and provide analogues of relational converse and taking graphs of maps. Our constructions are functorial in the parameter choices, establishing mathematical connections between different application domains. We illustrate our techniques by constructing many existing models from the literature, and new models that open up ground for further development.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases Process Theory, Categorical Compositional Semantics, Generalized Relations, Hypergraph Category, Compact Closed Category

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.17

1 Introduction

The term "process theory" has recently been introduced [11] to describe compositional theories of abstract processes. These process theories typically consist of a graphical language for reasoning about composite systems, and a categorical semantics tailored to the application domain. This compositional perspective has been incredibly successful in reasoning about questions in quantum computation and quantum foundations. The scope of the process theoretic perspective encompasses many other application domains, including natural language processing [12], signal flow graphs [8], control theory [3], Markov processes [5], electrical circuits [4] and even linear algebra [31].

When considering a new application of the process theoretic approach, the question arises of how to find a suitable categorical setting capturing the phenomena of interest. Dagger compact closed categories are of particular importance as they have an elegant graphical calculus, and many of the examples cited above live in compact closed categories.

We illustrate the process of constructing new dagger compact closed categories with two examples in the theory of human cognition, as developed in [18, 19]. This is an unconven-

* This work was partially funded by the AFSOR grant "Algorithmic and Logical Aspects when Composing Meanings" and the FQXi grant "Categorical Compositional Physics".

tional application area, and therefore highlights clearly the challenges faced when trying to model a new problem domain in a process theoretic manner.

As our first example, we consider how the notion of convexity can be incorporated into a compact closed setting. Convexity is important in mathematical models of cognition, where it is argued that the meaningful concepts should be closed under forming mixtures. Informally, if we have a space representing animals, then if two points describe dogs, we would expect any points "in between" should also be models of the concept of being a dog.

An algebraic model of convexity is given by the Eilenberg-Moore algebras of the finite distribution monad. These algebras, referred to as convex algebras, are sets equipped with a well behaved operation for forming convex mixtures of elements. Informally, we denote such a convex mixture as $\sum_i p_i x_i$ where the p_i are positive reals summing to one, and the x_i are elements of the algebra. This notation is not intended to imply there are independent addition and scaling operations that can be applied to the individual elements.

The Eilenberg-Moore category of any monad on **Set**[1], is itself a regular category. Therefore the category of convex algebras is regular and we can form its category of relations, denoted **ConvexRel**. It is well known that the category of relations over a regular category is a dagger compact closed category [20]. Concretely, a convex relation is an ordinary binary relation R which is closed under forming convex mixtures, in the sense that implications of the following form hold

$$R(a_1, b_1) \wedge ... \wedge R(a_n, b_n) \Rightarrow R(\sum_i p_i a_i, \sum_i p_i b_i) \tag{1}$$

A **state** of an object A in a monoidal category is a morphism of type $I \to A$ where I is the monoidal unit. The states in **ConvexRel** are the convex subsets, as we may have hoped. This model was used as the mathematical basis for a compositional model of cognition [7].

As our second example, we return to the mathematics of cognition. It is natural to think about notions of nearness and distance for models of reasoning, a wolf is nearly a dog, a squirrel is closer to being a rat than an elephant. We would therefore like to capture metrics within our model. We now consider how to introduce metrics into a compact closed setting. The construction used in the previous example is not applicable as the various natural categories of metric spaces are not regular. Therefore, we will require a new approach, which will entail a small detour. We begin by introducing the notion of a quantale.

▶ **Definition 1** (Quantale)**.** A **quantale** is a join complete partial order Q with a monoid structure $(\otimes, k)$ satisfying the following distributivity axioms, for all $a, b \in Q$ and $A, B \subseteq Q$

$$a \otimes \left[\bigvee B\right] = \bigvee\{a \otimes b \mid b \in B\} \qquad \left[\bigvee A\right] \otimes b = \bigvee\{a \otimes b \mid a \in A\}$$

A quantale is said to be **commutative** if its monoid structure is commutative.

▶ **Example 2.** Every locale [22] is a commutative quantale, and in particular any complete chain is a commutative quantale with

$$\bigvee A = \sup A \qquad a_1 \otimes a_2 = \min(a, b) \qquad k = \top$$

The **Boolean quantale B** and the **interval quantale I** are the chains $\{0, 1\}$ and $[0, 1]$ of real numbers, with their usual orderings. The quantale **F** is given by the chain $[0, \infty]$ of

[1] In fact, any regular category in which every regular epimorphism has a section.

extended positive reals with the *reverse* ordering. An important example of a commutative quantale that does not correspond to a locale is the **Lawvere quantale C** with underlying set the extended positive reals with reverse order and algebraic structure

$$\bigvee A = \inf A \qquad a_1 \otimes a_2 = a_1 + a_2 \qquad k = 0$$

A binary relation between two sets A and B can be described by its characteristic function $A \times B \to 2$ where 2 is the two element set of Boolean truth values. We can generalize the notion of binary relation by allowing the truth values to be taken in a suitable choice of quantale Q, as a function of the form $A \times B \to Q$. We can see this as a potentially infinite matrix of truth values. These binary relations form a category $\mathbf{Rel}(Q)$, with identities and composition given by suitable generalizations of their matrix theoretic analogues. We will prove that if the quantale of truth values is commutative, $\mathbf{Rel}(Q)$ is in fact dagger compact closed. So we have found another dagger compact closed category, but what has this got to do with metrics? In order to establish the required connection, we note that we can order relations pointwise in the quantale order. This order structure makes $\mathbf{Rel}(Q)$ into a poset-enriched symmetric monoidal category. This means we can consider internal monads, in the sense of formal category theory [32]. These identify important "structured objects" within our categories. For example, internal monads in $\mathbf{Rel}(\mathbf{B})$ are preorders on their underlying set, and the internal monads of $\mathbf{Rel}(\mathbf{I})$ can be seen as a fuzzy generalization of the notion of a preorder. The key example is the internal monads of $\mathbf{Rel}(\mathbf{C})$. These are endo-relations such that $R(a,a) = 0$ and $R(a,b) + R(b,c) \geq R(a,c)$, that is, they are *generalized metric spaces* [25]. The internal monads of $\mathbf{Rel}(\mathbf{F})$ are similar, satisfying $R(a,a) = 0$ and $\max(R(a,b), R(b,c)) \geq R(a,c)$, and can be thought of as *generalized ultrametric spaces*. So in particular, $\mathbf{Rel}(\mathbf{C})$ gives us a partial order enriched dagger compact closed category in which the internal monads are generalized metric spaces. Such categories of relations have been proposed as a unifying categorical setting for investigating various topological notions, see [10, 21]. Multi-valued relations have also been investigated for compositional models of natural language [13].

To recap, we have constructed two compact closed categories using differing techniques that can be found in the literature. Firstly, by exploiting relations respecting algebraic structure, standard monad and regular category theory provided us with a category where the states are exactly convex subsets. Secondly, generalizing the notion of relations in a different direction, we produced a category where the internal monads are generalized metric spaces. So, using rather ad-hoc methods, we have solved two modelling problems using generalizations of binary relations. This prompts several questions:

- How do these constructions relate to each other? In particular, can we simultaneously work with convexity and metrics in an appropriate setting? Can they be seen as instances of a general construction?
- Does the notion of binary relation permit further axes of variation, producing additional examples of compact closed categories? As these parameters vary, can the resulting categories be related? Formally, this is a question of functoriality in a suitable sense.

These questions provide the starting point for our investigations. We also observe that the categories we identified in our examples are both in fact instances of Fong and Kissinger's hypergraph categories [16]. These are a particularly well behaved class of dagger compact closed categories, and this will be our technical setting for the remainder of the paper. We summarize our contributions as follows

- We provide parameterized constructions of hypergraph categories of generalized relations and spans in Theorems 13 and 19, and show they have analogues of relational converse

and taking the graph of an underlying morphism. Many further aspects are shown to commute with this important structure. In Section 5 the resulting categories are shown to be appropriately order enriched.

- We address questions of functoriality. Theorem 29 shows that generalized spans can be functorially mapped to generalized relations. Section 6 shows that homomorphisms of truth values functorially induce functors between models. In Section 7 we show that our constructions are functorial in the choices of algebraic structure. We also describe how the algebraic and truth value structures interact, providing connections with notions of resource sensitivity in the sense of linear logic. Finally, in Theorem 47 we show that the functors induced by changes of parameters commute with each other.
- Our methods give explicit concrete descriptions of the mathematical objects of interest, suitable for use in applications. We provide many examples illustrating the flexibility of our techniques, particularly to the construction of new and existing models of natural language processing and cognition applications.

Related Work

Categories of relations have been studied in the form of allegories [17]. This work is somewhat removed from our approach, the heavy use of the modular law does not directly yield the graphical phenomena of interest. Of more direct relevance is the concept of cartesian bicategory of [9]. Although graphical notation is not used directly in this work, these categories can be seen as close relatives of the hypergraph categories resulting from our constructions. The emphasis in the study of cartesian bicategories was characterization rather than construction of models.

A somewhat syntactic approach to constructing categories with graphical calculi is the use of PROPs [26, 24]. They have recently been used to construct various categorical models relating to control theory [8, 33, 14]. These methods begin with syntax and equations, and freely derive a resulting category. This style is most effective when the application under consideration has well understood calculational properties. Our approach instead emphasizes the direct construction of models which can then be investigated for their suitability to a given application.

The beautiful work on decorated cospans and corelations of [15, 16], motivated by the program of network theory initiated in [2], is of most direct relevance to our approach. In a precise sense, the decorated corelation construction is completely generic, every hypergraph category is produced by that construction. Our emphasis is different, we do not aim for maximum generality. Instead, our aim is conceptually motivated parameterization. By providing four clearly motivated features that can be adjusted to application needs, we aim for a practical construction with which investigators using process theories can construct new models with desirable features.

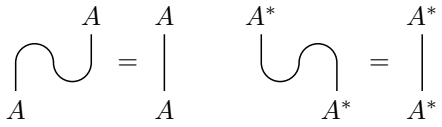
2 Mathematical Background

In this section we briefly establish some background. We will be interested in particular types of symmetric monoidal categories, and will make use of their corresponding graphical languages [30]. Technical background on monoidal categories and general categorical notions can be found in [27]. We will also refer to toposes and their internal languages in places, a standard reference is [23]. The paper has been written with the intention that it should be readable without any detailed knowledge of topos theory. For such readers, definitions should be read as if they pertain to ordinary sets, functions and predicate logic. We will

write **Set**, **Pos** and **Preord** for the categories of sets, partially ordered sets and preordered sets, with their usual homomorphisms.

▶ **Definition 3** (Compact Closed Category)**.** An object A in a symmetric monoidal category is said to have dual A^* if there exist unit $\eta : I \to A^* \otimes A$ and counit $\epsilon : A \otimes A^* \to I$ morphisms. These morphisms are depicted in the graphical calculus using special notation of Diagram (2) and are required to satisfy the **snake equations** of Diagram (3). [2]

$$\epsilon : \quad \eta : \tag{2}$$

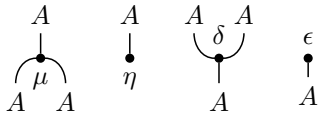


(3)

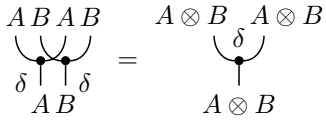
A **compact closed category** is a symmetric monoidal category in which every object has a dual. A compact closed category $\mathcal{A}$, equipped with an identity on objects involution $(-)^\dagger : \mathcal{A}^{op} \to \mathcal{A}$ coherent with the symmetric monoidal compact closed structure, is referred to as a **dagger compact closed category** [1].

▶ **Example 4.** The canonical example of a dagger compact closed category of relevance to the current work is the category **Rel** of sets and binary relations between them. The symmetric monoidal structure is given by cartesian products of sets, and the dagger by the usual converse of relations. Objects are self-dual, with the unit on a set A given by the relation $\{(*, (a, a)) \mid a \in A\}$ and the counit as its converse.

▶ **Definition 5** (Hypergraph Category)**.** A **hypergraph category** is a symmetric monoidal category such that every object A carries both a commutative monoid (η, μ) and a cocommutative comonoid (ϵ, δ) structure, as depicted in Diagram (4), satisfying the coherence conditions of Diagram (5), and their obvious dual [16].

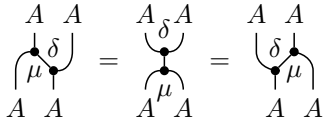


(4)



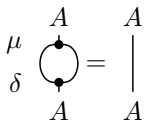
(5)

Here, we overload the use of the symbols μ, η, δ and ϵ to avoid cluttering our diagrams with indices or subscripts. We will exploit similar overloading of names in many places in what follows. The morphisms μ and δ must also satisfy the Frobenius (6) and special (7) axioms:



(6)

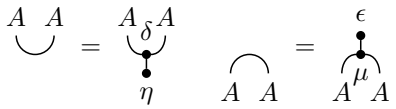
[2] Our string diagrams are oriented bottom to top.



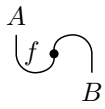
(7)

Our interest in hypergraph categories is that they are a particularly pleasant form of dagger compact closed category, as established by the following well known observation.

▶ **Proposition 6.** *Every hypergraph category is a dagger compact closed category, with the cup and cap given by Equation* (8) *and the dagger of a morphism* $f : A \to B$ *given by its* **transpose**, *shown in Diagram* (9).



(8)



(9)

▶ **Example 7.** The category **Rel** is also an example of a hypergraph category. The cocommutative comonoid is given by the relations

$$\epsilon = \{(a, *) \mid a \in A\} \qquad \delta = \{(a, (a, a)) \mid a \in A\}$$

The monoid is the relational converse of the comonoid structure. The induced dagger compact closed structure of Proposition 6 is exactly that described in Example 4.

As a final technical point, we will be working with various categories with finite products. Throughout, we will implicitly assume a choice of terminal object and binary products has been given. To reduce clutter, we therefore resist repeating this assumption in the statements of our subsequent theorems.

3 Relations

The aim in this section is to broadly generalize the notion of binary relation between sets, in order to support our motivating examples, and to provide scope for many other variations. We observed, for sets A and B, and quantale Q, that we can consider a function $A \times B \to Q$ as a relation, with truth values taken in the quantale. For such generalized relations, we define identities[3] and composition of relations $R : A \to B$ and $S : B \to C$ by analogy with the usual notions:

$$1_A(a_1, a_2) = \bigvee\{k \mid a_1 = a_2\}$$

$$(S \circ R)(a, c) = \bigvee\{R(a, b) \otimes S(b, c) \mid b \in B\}$$

We then observe that all of these definitions actually make sense in the internal language of an arbitrary topos. This leads us to the following definition.

[3] Our definition of identities is suitable for interpretation in the internal logic of a topos, unlike the more natural definition by cases.

▶ **Definition 8** (Q-relation)**.** Let $\mathcal{E}$ be a topos, and Q an internal quantale. A **Q-relation** between $\mathcal{E}$ objects A and B is an $\mathcal{E}$-morphism of type $A \times B \to Q$. $\mathcal{E}$-objects and Q-relations between them form a category $\mathbf{Rel}(Q)$, with identities and composition as described above.

Definition 8 is a first step in the right direction, but in order to capture convexity, as discussed in the introduction, we must find a way of incorporating algebraic structure. If we consider an algebraic signature (Σ, E) with set of operations Σ and equations E, the general form of Equation (1), for n-ary operation $\sigma \in \Sigma$, is

$$R(a_1, b_1) \wedge ... \wedge R(a_n, b_n) \Rightarrow R(\sigma(a_1, ..., a_n), \sigma(b_1, ..., b_n))$$

We will require throughout that all operation symbols have finite arity, as is conventional in universal algebra.

It is then natural to consider replacing the logical components of this definition with the structure of our chosen quantale. This leads to the definition we require.

▶ **Definition 9** (Algebraic Q-relation)**.** Let $\mathcal{E}$ be a topos, and Q an internal quantale. Let (Σ, E) be an algebraic variety in $\mathcal{E}$. An **algebraic** Q-relation between (Σ, E)-algebras A and B is a Q-relation between their underlying $\mathcal{E}$-objects such that for each $\sigma \in \Sigma$ the following axiom holds

$$R(a_1, b_1) \otimes ... \otimes R(a_n, b_n) \leq R(\sigma(a_1, ..., a_n), \sigma(b_1, ..., b_n)) \tag{10}$$

(Σ, E)-algebras and algebraic Q-relations form a category $\mathbf{Rel}_{(\Sigma,E)}(Q)$, with identities and composition as for their underlying Q-relations.

There is some subtlety to the interaction between truth values and algebraic structure, we will return to this topic in Section 7. We now continue studying the categorical structure of algebraic Q-relations.

▶ **Proposition 10.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal commutative quantale. The category $\mathbf{Rel}_{(\Sigma,E)}(Q)$ is a symmetric monoidal category. The symmetric monoidal structure is inherited from the finite products in $\mathcal{E}$.*

The notions of taking the converse of a relation, and taking the graph of an underlying function generalize smoothly to algebraic Q-relations, in a manner that respects all the relevant categorical structure.

▶ **Proposition 11.** *[Converse and Graph] Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal commutative quantale. There are identity on objects strict symmetric monoidal* **converse** *and* **graph** *functors with actions on morphisms:*

$$(-)^\circ : \mathbf{Rel}_{(\Sigma,E)}(Q)^{op} \to \mathbf{Rel}_{(\Sigma,E)}(Q)$$
$$R^\circ(b, a) = R(a, b)$$

$$(-)_\circ : \mathbf{Alg}(\Sigma, E) \to \mathbf{Rel}_{(\Sigma,E)}(Q)$$
$$f_\circ(a, b) = \bigvee \{k \mid f(a) = b\}$$

The symmetric monoidal structure on $\mathbf{Alg}(\Sigma, E)$ is the finite product structure.

The graph functor allows us to lift structures from the underlying category of algebras. The following canonical comonoids are of particular conceptual importance.

▶ **Proposition 12.** *Let $\mathcal{E}$ be a category with finite products. Each object A carries a cocommutative comonoid structure satisfying the coherence equations* (5)*, via the canonical morphisms* $! : A \to 1$ *and* $\langle 1_A, 1_A \rangle : A \to A \times A$*.*

Finally, we are in a position to establish that our categories of algebraic Q-relations are hypergraph categories.

▶ **Theorem 13.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal commutative quantale. The category $\mathbf{Rel}_{(\Sigma,E)}(Q)$ is a hypergraph category. The cocommutative comonoid structure is given by the graphs of the canonical comonoids described in Proposition 12, and the monoid structure is given by their converses.*

We quickly return to one of the examples discussed in the introduction.

▶ **Example 14.** The convex algebras discussed in the introduction can be presented by a family of binary operations $+^p$, where $p \in (0, 1)$, for forming pairwise convex combinations satisfying suitable equations. Writing Convex for this signature, we can construct **ConvexRel** as $\mathbf{Rel}_{\mathrm{Convex}}(\mathbf{B})$, where $\mathbf{B}$ is the Boolean quantale.

4 Spans

Generalizing the truth values, algebraic structure and ambient category has provided three degrees of freedom for describing custom hypergraph categories. Currently we can vary the underlying topos, quantale and choice of algebraic structure. We now investigate a fourth, final direction of variation.

If we consider a span of sets $A \xleftarrow{f} X \xrightarrow{g} B$, we can consider an element $x \in X$ as a proof witness relating $f(x)$ and $g(x)$. Spans $A \xleftarrow{f} X \xrightarrow{g} B$ and $B \xleftarrow{h} Y \xrightarrow{k} C$ are composed by pulling back g along h. Recall that in **Set** this pullback is given explicitly by $\{(x, y) \mid g(x) = h(y)\}$, with the obvious projections. Therefore, a pair (x, y) relates a and c exactly if x relates a to some b and this b is related to c by y. So, at least for the category **Set**, we can think of spans as proof relevant relations. This is the intuition we now pursue, starting by adjusting the notion of Q-relation in Definition 8 to the setting of spans.

▶ **Definition 15** (Q-span)**.** Let $\mathcal{E}$ be a finitely complete category, and Q an internal monoid. A **Q-span** of type $A \to B$ is a quadruple (X, f, g, χ) where $(X, f : X \to A, g : X \to B)$ is a span in $\mathcal{E}$ and $\chi : X \to Q$ is a $\mathcal{E}$-morphism, referred to as the **characteristic morphism**. Two Q-spans $(X, f, g, \chi), (Y, h, k, \xi)$ are composed by composing their underlying spans by pullback, and taking the resulting characteristic morphism to be

$$X \times_C Y \xrightarrow{\langle p_1, p_2 \rangle} X \times Y \xrightarrow{\chi \times \xi} Q \times Q \xrightarrow{\otimes} Q$$

where p_1 and p_2 are the pullback projections.

A **morphism of Q-spans** $\alpha : (X_1, f_1, g_1, \chi_1) \to (X_2, f_2, g_2, \chi_2)$ between two Q-spans of type $A \to B$ is a $\mathcal{E}$-morphism $\alpha : X_1 \to X_2$ such that

$$f_1 = f_2 \circ \alpha \qquad g_1 = g_2 \circ \alpha \qquad \chi_1 = \chi_2 \circ \alpha$$

▶ Remark. When discussing Q-spans in the remainder of this paper, we actually intend isomorphism classes of spans with respect to the homomorphisms of Definition 15. This convention is common when considering categories of ordinary spans, where composition of spans via pullback is only defined up to isomorphism. All definitions and calculations using

representatives will respect this isomorphism structure. These isomorphism classes of Q-spans form a category $\mathbf{Span}(Q)$. If we write χ_k for the constant morphism $\chi_k = A \xrightarrow{!} 1 \xrightarrow{k} Q$ then the identity at A is given by the Q-span $(A, 1, 1, \chi_k)$.

The key step now is to incorporate algebraic structure into the picture, paralleling the ideas of Definition 9. In this case, things are slightly more complicated as we have to explicitly administer the proof witnesses in the spans. We also must introduce an ordering on our truth values in order to specify the necessary axiom.

▶ **Definition 16.** Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. For (Σ, E)-algebras A and B, an **algebraic** Q-span is a quadruple (X, f, g, χ) which is a Q-span between the underlying $\mathcal{E}$-objects such that for every $\sigma \in \Sigma$ if

$$\bigwedge_i (f(x_i) = a_i \wedge g(x_i) = b_i)$$

then there exists x such that $f(x) = \sigma(a_1, ..., a_n)$ and $g(x) = \sigma(b_1, ..., b_n)$ and

$$\bigotimes_i \chi(x_i) \leq \chi(x)$$

(Σ, E)-algebras and algebraic Q-spans form a category $\mathbf{Span}_{(\Sigma,E)}(Q)$ with identities and composition given as for the underlying Q-spans.

As with the algebraic Q-relations in Section 3, we obtain a symmetric monoidal category with analogues of relational converse and taking graphs.

▶ **Proposition 17.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. The category $\mathbf{Span}_{(\Sigma,E)}(Q)$ is a symmetric monoidal category. The symmetric monoidal structure is inherited from the finite product structure in $\mathcal{E}$.*

▶ **Proposition 18.** *[Converse and Graph] Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. There are identity on objects strict symmetric monoidal* **converse** *and* **graph** *functors with actions on morphisms:*

$$(-)^\circ : \mathbf{Span}_{(\Sigma,E)}(Q)^{op} \to \mathbf{Span}_{(\Sigma,E)}(Q)$$
$$(X, f, g, \chi)^\circ = (X, g, f, \chi)$$

$$(-)_\circ : \mathbf{Alg}(\Sigma, E) \to \mathbf{Span}_{(\Sigma,E)}(Q)$$
$$f_\circ = (A, 1, f, \chi_k)$$

As before, we can exploit the graph construction and the canonical comonoids of Proposition 12 to establish the existence of a hypergraph structure.

▶ **Theorem 19.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. The category $\mathbf{Span}_{(\Sigma,E)}(Q)$ is a hypergraph category. The cocommutative comonoid structure is given by the graphs of the canonical comonoids described in Proposition 12, and the monoid structure is given by their converses.*

This construction presents new possibilities, that can be combined with other features, opening fresh directions for investigation that may not have been immediately apparent.

▶ **Example 20.** The span construction allows us to build variations on the models we are already interested in. For example, we can now consider a proof relevant version of the model in Example 14. From a practical perspective, this presents the possibility of models in which we can describe the interaction of cognitive phenomena, and provide quantitative evidence for any relationships that we conclude hold.

▶ **Example 21.** Instead of using **Set** as our base topos in our models, we could consider using a presheaf topos $[\mathcal{C}^{op}, \mathbf{Set}]$ for a small category $\mathcal{C}$. This allows us to construct models using "sets varying with context", incorporating all the features discussed in the previous examples. In linguistic or cognitive examples, contexts could describe time, the agents involved or the broader setting in which meaning should be interpreted. These context sensitive models present a lot of new expressive potential, and will be investigated in detail in future work.

5 Order Enrichment

In order to meaningfully discuss internal monads, we require some 2-categorical structure on our relational constructions. Specifically, we introduce an appropriate ordering on our morphisms. Order enrichment is also important from a practical perspective when modelling real world applications. For example, in natural language applications, we are often interested in phenomena such as ambiguity [29, 28] and lexical entailment [6], and these are best studied from an order theoretic perspective.

Generalizing the situation for ordinary set theoretic binary relations, we introduce an ordering on Q-relations.

▶ **Definition 22.** Let $\mathcal{E}$ be a topos and Q an internal quantale. We define a partial order on Q-relations as follows

$$R \subseteq R' \quad \text{iff} \quad \forall a, b. R(a,b) \leq R'(a,b)$$

Algebraic Q-relations are ordered similarly, by comparing their underlying Q-relations.

▶ **Theorem 23.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal commutative quantale. The category $\mathbf{Rel}_{(\Sigma,E)}(Q)$ is a partially ordered symmetric monoidal category.*

Q-spans can also be ordered, in a manner analogous to that for relations, but explicitly taking into account the proof witnesses.

▶ **Definition 24.** For topos $\mathcal{E}$ and internal partially ordered monoid Q, we define a preorder on Q-spans by saying $(X_1, f_1, g_1, \chi_1) \subseteq (X_2, f_2, g_2, \chi_2)$ if there is a $\mathcal{E}$-monomorphism $m : X_1 \to X_2$ such that $f_1 = f_2 \circ m$, $g_1 = g_2 \circ m$ and $\forall x. \chi_1(x) \leq \chi_2(m(x))$. Algebraic Q-spans are ordered similarly, by comparing their underlying Q-spans.

▶ **Theorem 25.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. The category $\mathbf{Span}_{(\Sigma,E)}(Q)$ is a preorder enriched symmetric monoidal category.*

The orders are respected by the important converse operation

▶ **Proposition 26.** *Let $\mathcal{E}$ be a topos, and (Σ, E) a variety in $\mathcal{E}$. If Q is an internal quantale, the converse functor of Proposition 11 is a partially ordered functor. If Q is an internal partially order monoid, the converse functor of Proposition 18 is a preordered functor.*

The order enrichment of Q-relations and Q-spans is crucial for us to be able to consider the internal monads central to the second example of the introduction.

▶ **Example 27.** The model incorporating metric spaces as internal monads, as discussed in the introduction, can be constructed with base topos **Set**, the empty algebraic signature and using the Lawvere quantale **C** as the choice of truth values.

Now that we have both algebraic and order structure available to us within the same construction, we can consider combining the features we are interested in, by making appropriate choices for the parameters used in the construction.

▶ **Example 28** (Convexity and Metrics)**.** We now see that we can combine *both* the convex and metric features in a single model. With underlying topos **Set**, we take our algebraic structure as in Example 14 and our quantale **C** as in Example 27. In this case we find the internal monads are distance measures $d : A \times A \to [0, \infty]$ satisfying the conditions

$$d(a,a) = 0 \qquad d(a,b) + d(b,c) \geq d(a,c) \qquad d(a_1,a_2) + d(b_1,b_2) \geq d(a_1 +^p b_1, a_2 +^p b_2)$$

These are generalized metric spaces that respect convex structure. The usual metric on $\mathbb{R}^n$ is an example of such a metric.

We can now consider the simplest aspect of functors induced by changes of parameters, the binary choice between proof relevance and provability. The next theorem shows that the orders on relations and spans are compatible, in the sense that we can collapse spans to relations using the join of the quantale to choose optimal truth values, and this mapping is functorial and respects the order structure.

▶ **Theorem 29.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$ and Q an internal commutative quantale. There is an identity on objects, strict symmetric monoidal* **Preord***-functor V, with action on morphisms:*

$$V : \mathbf{Span}_{(\Sigma,E)}(Q) \to \mathbf{Rel}_{(\Sigma,E)}(Q)$$
$$V(X, f, g, \chi)(a, b) = \bigvee \{\chi(x) \mid f(x) = a \wedge g(x) = b\}$$

V commutes with graphs and converses in that $V \circ (-)_\circ = (-)_\circ$ and $(-)^\circ \circ V^{op} = V \circ (-)^\circ$.

6 Changing Truth Values

We would expect that homomorphisms between our structures of truth values lead to functorial relationships between models. This all goes through very smoothly, as we now elaborate. Firstly, for algebraic Q-relations, it is natural to consider internal quantale homomorphisms.

▶ **Theorem 30.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and $h : Q_1 \to Q_2$ a morphism of internal commutative quantales. There is an identity on objects, strict symmetric monoidal* **Pos***-functor $h^* : \mathbf{Rel}_{(\Sigma,E)}(Q_1) \to \mathbf{Rel}_{(\Sigma,E)}(Q_2)$, with action on morphisms $R \mapsto h \circ R$. The assignment $h \mapsto h^*$ is functorial.*

In the case of the span constructions, morphisms of partially ordered monoids are the appropriate notion of homomorphism to consider.

▶ **Theorem 31.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and $h : Q_1 \to Q_2$ a morphism of internal partially ordered commutative monoids. There is an identity on objects, strict symmetric monoidal* **Preord***-functor $h^* : \mathbf{Span}_{(\Sigma,E)}(Q_1) \to \mathbf{Span}_{(\Sigma,E)}(Q_2)$, with action on morphisms $(X, f, g, \chi) \mapsto (X, f, g, h \circ \chi)$. The assignment $h \mapsto h^*$ is functorial.*

Both these functors commute with graphs and converses.

▶ **Proposition 32.** *With the same assumptions, the induced functors of Theorems 30 and 31 commute with graphs and converses. That is, $h^* \circ (-)_\circ = (-)_\circ$ and $(-)^\circ \circ (h^*)^{op} = h^* \circ (-)^\circ$.*

▶ **Example 33.** For any commutative quantale Q there is a partially ordered monoid morphism $1 \to Q$, induced by the monoid unit. Here, 1 is the terminal quantale. Therefore there is a strict symmetric monoidal functor $\mathbf{Span}_{(\Sigma,E)}(1) \to \mathbf{Span}_{(\Sigma,E)}(Q)$. This example motivates our use of partially ordered monoids, rather than simply restricting to the quantales of interest in our primary applications, as the required morphism is not a quantale morphism.

▶ **Example 34.** There is a quantale morphism $\mathbf{B} \to \mathbf{C}$ from the Boolean to the Lawvere quantale. The induced functor identifies the ordinary binary relations as living within the category $\mathbf{Rel}(\mathbf{C})$ that we introduced to capture metric spaces as internal monads.

When using ordinary relational models of natural language, the meanings of two sentences are typically compared using the inner product of the corresponding states. This is a crude measure of similarity as it is a simple Boolean test of overlap. By embedding Boolean relations into $\mathbf{Rel}(\mathbf{C})$, more subtle comparisons can be made using a metric. For example we can measure how far apart two states are at their nearest point.

7 Algebraic Structure

We now investigate the interaction between truth values and algebraic structure. Again, this will lead to functorial relationships between models, but the subject is more delicate than in the previous sections. The essential detail is that in Equation (10) is only required to hold for the operations in our signature. It does not directly say anything about derived terms and operations. We will require several definitions in order to make the situation precise.

▶ **Definition 35.** Let (Σ, E) be an algebraic signature. We say that a term τ over a finite set of variables is **affine** if it uses each variable at most once and **relevant** if it uses each variable at least once. A term is **linear** if it is both affine and relevant. We will refer to a term as **cartesian** to emphasize that its use of variables is unrestricted. We use the same terminology for the derived operation associated to τ. An **interpretation** of signature (Σ_1, E_1) in signature (Σ_2, E_2) is a mapping assigning each $\sigma \in \Sigma_1$ to a derived term of (Σ_2, E_2) of the same arity, such that the equations E_1 can be proved in equational logic from E_2. We say that an interpretation is **linear**, **affine**, **relevant** or **cartesian** if all the derived terms used in the interpretation are suitably restricted. It is standard that every interpretation i contravariantly induces a functor $\hat{i}$ between the categories of algebras.

▶ **Definition 36.** Let $\mathcal{E}$ be a topos. If Q is an internal quantale, we say that a Q-relation R is **affine** if $R(a_1, b_1) \otimes R(a_2, b_2) \leq R(a_1, b_1)$ and **relevant** if $R(a, b) \leq R(a, b) \otimes R(a, b)$. R is **cartesian** if it is both affine and linear. We say that R is **linear** to emphasize that no additional axioms are assumed to hold. Similarly, if Q is an internal partially ordered monoid, we say that a Q-span (X, f, g, χ) is **affine** if $\chi(x_1) \otimes \chi(x_2) \leq \chi(x_1)$, and **relevant** if $\chi(x) \leq \chi(x) \otimes \chi(x)$. A Q-span is said to be **cartesian** if it is both affine and relevant, and **linear** if no additional axioms are assumed to hold.

Our terminology is derived from that sometimes used for variants of linear logic. If we view truth values as resources, the question is when can these resources be "copied" or "deleted". The next proposition shows that if our truth values are well behaved, so are our morphisms.

▶ **Lemma 37.** *Let Q be an internal quantale. If $p \otimes q \leq p$ holds, every relation is affine and if $p \leq p \otimes p$ then every relation is relevant. Similarly, if Q is an internal partially ordered monoid, if $p \otimes q \leq p$ holds, every span is affine and if $p \leq p \otimes p$ then every span is relevant.*

Each of our special classes of relations and spans forms a hypergraph category.

▶ **Theorem 38.** *Let $\mathcal{E}$ be a topos and (Σ, E) a variety in $\mathcal{E}$. If Q is a commutative quantale, the affine, relevant and cartesian relations each form a sub-hypergraph category of $\mathbf{Rel}_{(\Sigma,E)}(Q)$. If Q is a commutative partially ordered monoid, the affine, relevant and cartesian spans each form a sub-hypergraph category of $\mathbf{Span}_{(\Sigma,E)}(Q)$. In each case, the morphisms in the image of the graph functor are all cartesian.*

▶ **Definition 39.** We will write $\mathbf{Rel}^{\mathrm{cart}}_{(\Sigma,E)}(Q)$ and $\mathbf{Span}^{\mathrm{cart}}_{(\Sigma,E)}(Q)$ for the sub-hypergraph categories of cartesian relations and spans described in Theorem 38, and use similar notation for the other restricted classes of morphisms.

Our restricted classes of relations respect the corresponding classes of derived terms.

▶ **Proposition 40.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$ and Q an internal commutative quantale. For linear (affine, relevant, cartesian) algebraic Q-relation $R : A \to B$ the axiom*

$$R(a_1, b_1) \otimes ... \otimes R(a_n, b_n) \leq R(\tau(a_1, ..., a_n), \tau(b_1, ..., b_n))$$

holds for every linear (affine, relevant, cartesian) n-ary derived operation τ.

Spans with sufficient structure also respect the corresponding types of derived terms.

▶ **Proposition 41.** *Let $\mathcal{E}$ be a topos, (Σ, E) a variety in $\mathcal{E}$, and Q an internal partially ordered commutative monoid. For (Σ, E)-algebras A and B, and linear (affine, relevant, cartesian) algebraic Q-span (X, f, g, χ) and n-ary linear (affine, relevant, cartesian) term τ if $\bigwedge_i (f(x_i) = a_i \wedge g(x_i) = b_i)$ then there exists x such that $f(x) = \tau(a_1, ..., a_n)$, $g(x) = \tau(b_1, ..., b_n)$ and $\bigotimes_i \chi(x_i) \leq \chi(x)$.*

Finally, we can establish a contravariant functorial relationship between interpretations and functors between relational models.

▶ **Theorem 42.** *Let $\mathcal{E}$ be a topos and Q an internal commutative quantale. Let $i : (\Sigma_1, E_1) \to (\Sigma_2, E_2)$ be a linear interpretation of signatures. There is an identity on morphisms strict symmetric monoidal functor $i^* : \mathbf{Rel}^{\mathrm{lin}}_{(\Sigma_2,E_2)}(Q) \to \mathbf{Rel}^{\mathrm{lin}}_{(\Sigma_1,E_1)}(Q)$, sending each (Σ_2, E_2)-algebra to the corresponding (Σ_1, E_1)-algebra under the interpretation. The assignment $i \mapsto i^*$ extends to a contravariant functor. Similar results hold for affine, relevant and cartesian interpretations and relations. In each case, the induced functors commute with graphs and converses. That is, $(-)_\circ \circ \hat{i} = i^* \circ (-)_\circ$ and $(-)^\circ \circ (i^*)^{op} = i^* \circ (-)^\circ$, where $\hat{i}$ is the interpretation induced functor of Definition 35.*

A similar contravariant functorial relationship holds between interpretations and functors between span based models.

▶ **Theorem 43.** *Let $\mathcal{E}$ be a topos and Q an internal partially ordered commutative monoid. Let $i : (\Sigma_1, E_1) \to (\Sigma_2, E_2)$ be a linear interpretation of signatures. There is an identity*

on morphisms strict monoidal functor $i^* : \mathbf{Span}^{\mathrm{lin}}_{(\Sigma_2,E_2)}(Q) \to \mathbf{Span}^{\mathrm{lin}}_{(\Sigma_1,E_1)}(Q)$*, sending each* (Σ_2, E_2)*-algebra to the corresponding* (Σ_1, E_1)*-algebra under the interpretation. The assignment* $i \mapsto i^*$ *extends to a contravariant functor. Similar results hold for affine, relevant and cartesian interpretations and spans. In each case, the induced functors commute with graphs and converses. That is,* $(-)_\circ \circ \hat{i} = i^* \circ (-)_\circ$ *and* $(-)_\circ \circ (i^*)^{op} = i^*(-)_\circ$ *where* $\hat{i}$ *is the interpretation induced functor of Definition 35.*

We also note that the extensional collapse functor of Theorem 29 also respects our different classes of spans and relations.

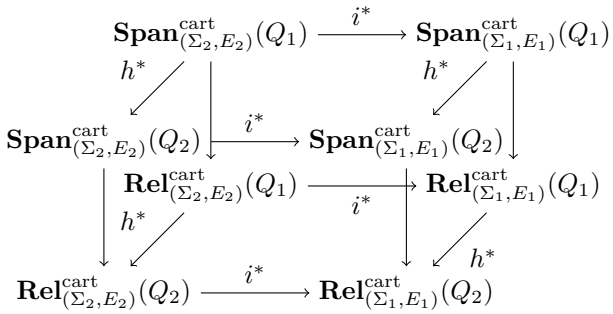
▶ **Proposition 44.** *Let* $\mathcal{E}$ *be a topos,* (Σ, E) *a variety in* $\mathcal{E}$ *and* Q *an internal commutative quantale. The functor* V *of Theorem 29 maps cartesian, affine and linear algebraic* Q*-spans to the corresponding class of algebraic* Q*-relations.*

▶ **Example 45.** Let $(\emptyset, \emptyset)$ denote the signature with no operations or equations. For any signature (Σ, E) there is a trivial *linear* interpretation $(\emptyset, \emptyset) \to (\Sigma, E)$. We therefore have, for every choice of internal quantale Q, strict symmetric monoidal forgetful functors $\mathbf{Rel}_{(\Sigma,E)}(Q) \to \mathbf{Rel}_{(\emptyset,\emptyset)}(Q)$ and $\mathbf{Span}_{(\Sigma,E)}(Q) \to \mathbf{Span}_{(\emptyset,\emptyset)}(Q)$.

▶ **Example 46.** The signature for convex algebras has linear interpretations in both real vector spaces and affine semilattices. Therefore for any commutative quantale Q, we find relations between real vector spaces and relations between affine join semilattices as sub-hypergraph categories of $\mathbf{Rel}_{Convex}(Q)$.

Finally, we establish that our various induced functors between models are independent, in that they all commute with each other.

▶ **Theorem 47.** *Let* $\mathcal{E}$ *be a topos,* $h : Q_1 \to Q_2$ *a morphism of internal commutative quantales and* $i : (\Sigma_1, E_1) \to (\Sigma_2, E_2)$ *a cartesian interpretation. For the induced functors of Theorems 30, 31, 42 and 43, the following diagram commutes:*



where the vertical arrows are the V *functors of Theorem 29. Similar diagrams commute for affine, relevant and linear interpretations, relations and spans.*

8 Conclusion

We have developed a parameterized scheme for constructing order enriched hypergraph categories, by generalizing the notion of binary relation along four axes of variation: the ambient category, the truth values, the algebraic structure and the choice between proof relevance and provability. This construction provides a concrete, conceptually motivated approach for producing models of process theories when investigating new applications. As well as recovering many existing models, by allowing us to combine features, the framework

points to new settings in which features such as convexity, distances, contextual meaning and proof witnesses can be incorporated into a single model. Detailed exploration of these new models in linguistic and cognition applications is left to later work.

Acknowledgments. The authors would like to thank Bob Coecke, Ignacio Funke, Kohei Kishida and Martha Lewis for feedback and discussions. We would also like to thank the anonymous reviewers for their comments and suggestions.

References

1 Samson Abramsky and Bob Coecke. A categorical semantics of quantum protocols. In *Logic in Computer Science, 2004. Proceedings of the 19th Annual IEEE Symposium on*, pages 415–425. IEEE, 2004.

2 John C Baez. Network theory (part 1). mathematical blog entry. URL: `https://johncarlosbaez.wordpress.com/2011/03/04/network-theory-part-1/`.

3 John C Baez and Jason Erbele. Categories in control. *Theory and Applications of Categories*, 30(24):836–881, 2015.

4 John C Baez and Brendan Fong. A compositional framework for passive linear networks. *arXiv preprint arXiv:1504.05625*, 2015.

5 John C Baez, Brendan Fong, and Blake S Pollard. A compositional framework for Markov processes. *Journal of Mathematical Physics*, 57(3):033301, 2016.

6 Dea Bankova. Comparing meaning in language and cognition - p-hypononymy, concept combination, asymmetric similarity. Master's thesis, University of Oxford, 2015.

7 Josef Bolt, Bob Coecke, Fabrizio Genovese, Martha Lewis, Daniel Marsden, and Robin Piedeleu. Interacting conceptual spaces. In Dimitrios Kartsaklis, Martha Lewis, and Laura Rimell, editors, *Proceedings of the 2016 Workshop on Semantic Spaces at the Intersection of NLP, Physics and Cognitive Science, SLPCS@QPL 2016, Glasgow, Scotland, 11th June 2016.*, volume 221 of *EPTCS*, pages 11–19, 2016. `doi:10.4204/EPTCS.221.2`.

8 Filippo Bonchi, Pawel Sobocinski, and Fabio Zanasi. Full abstraction for signal flow graphs. *ACM SIGPLAN Notices*, 50(1):515–526, 2015.

9 Aurelio Carboni and Robert F C Walters. Cartesian bicategories I. *Journal of pure and applied algebra*, 49(1-2):11–32, 1987.

10 Maria Manuel Clementino and Walter Tholen. Metric, topology and multicategory–a common approach. *Journal of Pure and Applied Algebra*, 179(1):13–47, 2003.

11 Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes. A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017.

12 Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical foundations for distributed compositional model of meaning. Lambek festschrift. *Linguistic Analysis*, 36:345–384, 2010.

13 Matej Dostal and Mehrnoosh Sadrzadeh. Many valued generalised quantifiers for natural language in the DisCoCat model. Technical report, QMUL, 2016.

14 Jason Erbele. *Categories in Control: Applied PROPs*. PhD thesis, University of California Riverside, 2016.

15 Brendan Fong. Decorated cospans. *Theory and Applications of Categories*, 30(33):1096–1120, 2015.

16 Brendan Fong. *The Algebra of Open and Interconnected Systems*. PhD thesis, University of Oxford, 2016.

17 Peter J Freyd and Andre Scedrov. *Categories, allegories*, volume 39. Elsevier, 1990.

18 Peter Gärdenfors. *Conceptual spaces: The geometry of thought*. MIT press, 2004.

19 Peter Gärdenfors. *The geometry of meaning: Semantics based on conceptual spaces*. MIT Press, 2014.

20 Chris Heunen and Sean Tull. Categories of relations as models of quantum theory. In Chris Heunen, Peter Selinger, and Jamie Vicary, editors, Proceedings of the 12th International Workshop on *Quantum Physics and Logic,* Oxford, U.K., July 15-17, 2015, volume 195 of *EPTCS*, pages 247–261. Open Publishing Association, 2015. doi:10.4204/EPTCS.195.18.

21 Dirk Hofmann, Gavin J Seal, and Walter Tholen. *Monoidal Topology: A Categorical Approach to Order, Metric, and Topology*, volume 153. Cambridge University Press, 2014.

22 Peter T Johnstone. *Stone spaces*, volume 3. Cambridge University Press, 1986.

23 Peter T Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*, volume II. Oxford University Press, 2002.

24 Stephen Lack. Composing PROPs. *Theory and Applications of Categories*, 13(9):147–163, 2004.

25 F William Lawvere. Metric spaces, generalized logic, and closed categories. *Rendiconti del seminario matématico e fisico di Milano*, 43(1):135–166, 1973.

26 Saunders MacLane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71(1):40–106, 1965.

27 Saunders MacLane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer, 1998.

28 R. Piedeleu, D. Kartsaklis, B. Coecke, and M Sadrzadeh. Open system categorical quantum semantics in natural language processing. In Lawrence S. Moss and Pawel Sobocinski, editors, *6th Conference on Algebra and Coalgebra in Computer Science, CALCO 2015*, volume 35 of *LIPIcs*, pages 270–289. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.CALCO.2015.270.

29 Robin Piedeleu. Ambiguity in categorical models of meaning. Master's thesis, University of Oxford, 2014.

30 Peter Selinger. A survey of graphical languages for monoidal categories. In *New structures for physics*, pages 289–355. Springer, 2010.

31 Pawel Sobocinski. Graphical linear algebra. mathematical blog. URL: https://graphicallinearalgebra.net/.

32 Ross Street. The formal theory of monads. *Journal of Pure and Applied Algebra*, 2(2):149–168, 1972.

33 Fabio Zanasi. *Interacting Hopf Algebras: The Theory of Linear Systems*. PhD thesis, l'École Normale Supérieure de Lyon, 2015.

Proper Functors and their Rational Fixed Point

Stefan Milius*

Lehrstuhl für Theoretische Informatik, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Abstract

The rational fixed point of a set functor is well-known to capture the behaviour of finite coalgebras. In this paper we consider functors on algebraic categories. For them the rational fixed point may no longer be a subcoalgebra of the final coalgebra. Inspired by Ésik and Maletti's notion of proper semiring, we introduce the notion of a proper functor. We show that for proper functors the rational fixed point is determined as the colimit of all coalgebras with a free finitely generated algebra as carrier and it is a subcoalgebra of the final coalgebra. Moreover, we prove that a functor is proper if and only if that colimit is a subcoalgebra of the final coalgebra. These results serve as technical tools for soundness and completeness proofs for coalgebraic regular expression calculi, e.g. for weighted automata.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases proper functor, proper semiring, coalgebra, rational fixed point

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.18

1 Introduction

Coalgebras allow to model many types of systems within a uniform and conceptually clear mathematical framework [25]. One of the key features of this framework is *final semantics*; the final coalgebra provides a fully abstract domain of system behaviour (i.e. it identifies precisely the behaviourally equivalent states). For example, the standard coalgebraic modelling of deterministic automata (without restricting to finite state sets) yields the set of formal languages as final coalgebra. Restricting to finite automata, one obtains precisely the regular languages [24]. It is well-known that this correspondence can be generalized to locally finitely presentable (lfp) categories [4], where *finitely presentable* objects play the role of finite sets. For a finitary functor F (modelling a coalgebraic system type) one then obtains the *rational fixed point* ϱF, which provides final semantics to all coalgebras with a finitely presentable carrier [17]. Moreover, the rational fixed point is fully abstract whenever the classes of finitely presentable and finitely generated objects agree in the base category and F preserves monomorphisms [7, Proposition 3.12]. While the latter assumption on F is very mild (and is not even needed in the case of a lifted set functor), the former one on the base category is more restrictive. However, it is still true for many categories used in the construction of coalgebraic system models (e.g. sets, posets, graphs, vector spaces, commutative monoids, nominal sets and convex sets).

In this paper we will consider rational fixed points in algebraic categories (a.k.a. finitary varieties), i.e. categories of algebras specified by a finitary signature of operation symbols and a set of equations (equivalently, these are precisely the Eilenberg-Moore categories for finitary monads on sets). Being the target of generalized determinization [28], these categories

* Supported by Deutsche Forschungsgemeinschaft (DFG) under project MI 717/5-1

provide a paradigmatic setting for coalgebraic modelling beyond sets. For example, non-deterministic automata, weighted or probabilistic ones [16], or context-free grammars [33] are coalgebraically modelled over the categories of join-semilattices, modules for a semiring, convex sets, and idempotent semirings, respectively. In algebraic categories one would like that the rational fixed point, in addition to being fully abstract, is determined already by those coalgebras carried by free finitely generated algebras, i.e. precisely those coalgebras arising by generalized determinization. In particular, this feature is used in completeness proofs for generalized regular expressions calculi [29, 28, 7]; there one proves that the quotient of syntactic expressions modulo axioms of the calculus is (isomorphic to) the rational fixed point by establishing its universal property as a final object for that quotient. A key feature of the settings in loc. cit. is that it suffices to verify the finality only w.r.t. coalgebras with a free finitely generated carrier.

The purpose of the present paper is to provide sufficient conditions on the algebraic base category and coalgebraic type functor that ensure such finality proofs are sound. More precisely, inspired by Ésik and Maletti's notion of a proper semiring (which is in fact a notion concerning weighted automata), we introduce *proper functors* (Definition 23), and we prove that for a proper functor on an algebraic category the rational fixed point is determined by the coalgebras with a free finitely presentable carrier. More precisely, let $T : \mathsf{Set} \to \mathsf{Set}$ be a finitary monad on sets and $F : \mathsf{Set}^T \to \mathsf{Set}^T$ be a finitary endofunctor preserving surjective T-algebra morphisms (note that the last assumption always holds if F is lifted from some endofunctor on Set). If F is proper, then the rational fixed point is the colimit φF of the inclusion functor of the full subcategory $\mathsf{Coalg}_{\mathsf{free}}\, F$ formed by all F-coalgebras of the form $TX \to FTX$, where X is a finite set (Theorem 27). Moreover, we show that a functor F is proper if and only if φF is a subcoalgebra of the final coalgebra νF (Theorem 26). As a consequence we also obtain that for a proper functor F finality of a given locally finitely presentable coalgebra can be established by only verifying that property for all coalgebras from $\mathsf{Coalg}_{\mathsf{free}}\, F$ (Corollary 29).

We also provide more easily established sufficient conditions on Set^T and F that ensure properness: F is proper if finitely generated algebras of Set^T are closed under kernel pairs and F maps kernel pairs to weak pullbacks in Set. For a lifting F this holds whenever the lifted functor on sets preserves weak pullbacks; in fact, in this case the above conditions were shown to entail Corollary 29 in previous work [7, Corollary 3.36]. However, the type functor (on the category of commutative monoids) of weighted automata with weights drawn from the semiring of natural numbers provides an example of a proper functor for which the above condition on Set^T fails.

Another recent related work concerns the so-called *locally finite fixed point* ϑF [19]; this provides a fully abstract behavioural domain whenever F is a finitary endofunctor on an lfp category preserving monomorphisms. In loc. cit. it was shown that ϑF captures a number of instances that cannot be captured by the rational fixed point, e.g. context free languages [33], constructively algebraic formal power-series [22, 34], Courcelle's algebraic trees [8, 2] and the behaviour of stack machines [15]. However, as far as we know, ϑF is not amenable to the simplified finality check mentioned above unless F is proper.

Putting everything together, in an algebraic category we obtain the following picture of fixed points of F (where $\twoheadrightarrow$ denotes quotient coalgebras and $\rightarrowtail$ a subcoalgebra):

$$\varphi F \twoheadrightarrow \varrho F \twoheadrightarrow \vartheta F \rightarrowtail \nu F. \tag{1}$$

We exhibit an example, where all four fixed points are different. However, if F is proper and preserves monomorphisms, then φF, ϱF and ϑF are isomorphic and fully abstract, i.e. they collapse to a subcoalgebra of the final one: $\varphi F \cong \varrho F \cong \vartheta F \rightarrowtail \nu F$.

The rest of the paper is structured as follows: in Section 2 we collect some technical preliminaries and recall the rational and locally finite fixed points more in detail. Section 3 introduces proper functors and presents all our results while in Section 4 we present the proof of our main result Theorem 26. Finally, Section 5 concludes the paper.

Due to space restrictions some proofs and details are omitted; these can be found in the full version of this paper [18].

2 Preliminaries

In this section we recall a few preliminaries needed for the subsequent development. We assume that readers are familiar with basic concept of category theory.

We denote the coproduct of two object X and Y of a category $\mathscr{A}$ by $X+Y$ with injections $\mathsf{inl} : X \to X + Y$ and $\mathsf{inr} : Y \to X + Y$.

▶ Remark 1. Recall that a *strong epimorphism* in a category $\mathscr{A}$ is an epimorphism $e : A \twoheadrightarrow B$ of $\mathscr{A}$ that has the unique diagonal property w.r.t. any monomorphism. More precisely, whenever we have a commutative square $m \cdot f = g \cdot e$, where $m : C \rightarrowtail D$ is a monomorphism, then there exists a unique diagonalization $d : B \to C$ with $d \cdot e = f$ and $m \cdot d = g$.

2.1 Algebras and Coalgebras

We assume that readers are familiar with algebras and coalgebras for an endofunctor. Given an endofunctor F on some category $\mathscr{A}$ we write $(\nu F, t)$ for the final F-coalgebra (if it exists). Recall, that the final F-coalgebra exists under mild assumptions on $\mathscr{A}$ and F, e.g. whenever $\mathscr{A}$ is locally presentable and F an accessible functor (see [4]). For any coalgebra $c : C \to FC$ we will write $\dagger c : C \to \nu F$ for the unique coalgebra morphism.

If $\mathscr{A}$ is a concrete category, i.e. equipped with a faithful functor $|\cdot| : \mathscr{A} \to \mathsf{Set}$, one defines *behavioural equivalence* as the following relation $\sim$: given two F-coalgebras (X, c) and (Y, d) then $x \sim y$ holds for $x \in |X|$ and $y \in |Y|$ if there is another F-coalgebra (Z, e) and F-coalgebra morphisms $f : X \to Z$ and $g : Y \to Z$ with $|f|(x) = |g|(y)$.

The base categories $\mathscr{A}$ of interest in this paper are the *algebraic categories*, i.e. categories of Eilenberg-Moore algebras (or T-algebras, for short) for a finitary monad T on Set. Equivalently, those categories are precisely the finitary varieties, i.e. category of Σ-algebras for a finitary signature Σ satisfying the a set of equations (e.g. the categories of monoids, groups, vector spaces, join-semilattices).

Given a monad T with unit $\eta : \mathsf{Id} \to T$ and multiplication $\mu : TT \to T$, we will sometimes make use of its *Kleisli extension*, i.e. the operation $(-)^\star$ that takes any morphism $f : X \to TY$ to $f^\star = \mu_Y \cdot Tf : TX \to TY$. Note that $f^\star$ is the unique T-algebra morphism from (TX, μ_X) to (TY, μ_Y) such that $f^\star \cdot \eta_X = f$.

▶ **Example 2.** The leading example in this paper are weighted automata considered as coalgebras. Let $(\mathbb{S}, +, \cdot, 0, 1)$ be a semiring, i.e. $(\mathbb{S}, +, 0)$ is a commutative monoid, $(\mathbb{S}, \cdot, 1)$ a monoid and the usual distributive laws hold: $r \cdot 0 = 0 = 0 \cdot r$, $r \cdot (s + t) = r \cdot s + r \cdot t$ and $(r + s) \cdot t = r \cdot t + s \cdot t$. We just write $\mathbb{S}$ to denote a semiring. As base category $\mathscr{A}$ we consider the category $\mathbb{S}$-Mod of $\mathbb{S}$-semimodules; recall that a (left) $\mathbb{S}$-*semimodule* is a commutative monoid $(M, +, 0)$ together with an action $\mathbb{S} \times M \to M$, written as juxtaposition sm for $r \in \mathbb{S}$ and $m \in M$, such that for every $r, s \in \mathbb{S}$ and every $m, n \in M$ the following laws hold:

$$\begin{array}{lll} (r+s)m = rm + sm & 0m = 0 & 1m = m \\ r(m+n) = rm + rn & r0 = 0 & r(sm) = (r \cdot s)m \end{array}$$

An $\mathbb{S}$-semimodule morphism is a monoid homomorphism $h\colon M_1 \to M_2$ such that $h(rm) = rh(m)$ for each $r \in \mathbb{S}$ and $m \in M_1$.

Now consider the functor $FX = \mathbb{S} \times X^A$ on $\mathbb{S}$-Mod, where A is an input alphabet. Then it is easy to see that an $\mathbb{S}$-weighted automaton with n states is precisely a coalgebra on the free $\mathbb{S}$-semimodule on n generators, i.e. $\mathbb{S}^n \to \mathbb{S} \times (\mathbb{S}^n)^A$. The final $\mathbb{S}$-coalgebra is carried by the set $\mathbb{S}^{A^*}$ of all *formal power series* (or *weighted languages*) over A with the obvious (coordinatewise) $\mathbb{S}$-semimodule structure and with the F-coalgebra structure given by $\langle o, t\rangle : \mathbb{S}^{A^*} \to \mathbb{S} \times (\mathbb{S}^{A^*})^A$ with $o(L) = L(\varepsilon)$ and $t(L)(a) = \lambda w.L(aw)$; it is straightforward to verify that o and t are $\mathbb{S}$-semimodule morphisms and form a final coalgebra.

An important special case of $\mathbb{S}$-weighted automata are ordinary nondeterministic automata. One takes $\mathbb{S} = \{0, 1\}$ the Boolean semiring for which the category of $\mathbb{S}$-semimodules is (isomorphic to) the category of join-semilattices. Then $FX = \{0, 1\} \times X^A$ is the coalgebraic type functor of deterministic automata with input alphabet A, and there is a bijective correspondence between an F-coalgebra on a free join-semilattice and non-deterministic automata. In fact in one direction one restricts $\mathcal{P}_\mathsf{f}X \to \{0, 1\} \times (\mathcal{P}_\mathsf{f}X)^A$ to the set X of generators, and in the other direction one performs the well-known subset construction. The final coalgebra is carried by the set of all formal languages on A in this case.

Another special case is where $\mathbb{S}$ is a field. In this case, $\mathbb{S}$-semimodules are precisely the vector spaces over the field $\mathbb{S}$. Moreover, since every field is freely generated by its basis, it follows that the $\mathbb{S}$-weighted automata are precisely those F-coalgebras whose carrier is a finite dimensional vector space over $\mathbb{S}$.

We will now recall a few properties of algebraic categories Set^T, where T is a finitary set monad, needed for our proofs.

▶ Remark 3.

1. Recall that every strong epimorphisms e in Set^T is regular, i.e. e is the coequalizer of some pair of T-algebra morphisms. It follows that the classes of strong and regular epimorphisms coincide, and these are precisely the surjective T-algebra morphisms.
2. We will later use that every free T-algebra TX is *(regular) projective*, i.e. given any surjective T-algebra morphism $q : A \twoheadrightarrow B$ then for every T-algebra morphism $h : TX \to B$ there exists a T-algebra morphism $g : TX \to A$ such that $q \cdot g = h$.
3. Furthermore, note that every finitely presentable T-algebra A is a regular quotient of a free T-algebra TX with a finite set X of generators. Indeed, A is presented by finitely many generators and relations. So by taking X as a finite set of generators of A, the unique extension of the embedding $X \hookrightarrow A$ yields a surjective T-algebra morphism $TX \twoheadrightarrow A$.

2.2 The Rational Fixed Point

As we mentioned in the introduction the canonical domain of behaviour of 'finite' coalgebras is the rational fixed point of an endofunctor on F. Its theory can be developed for every finitary endofunctor on a locally finitely presentable category. We will now recall the necessary background material.

A *filtered colimit* is the colimit of a diagram $\mathscr{D} \to \mathscr{C}$ where $\mathscr{D}$ is a filtered category (i.e. every finite subdiagram has a cocone in $\mathscr{D}$), and a *directed colimit* is a colimit whose diagram scheme $\mathscr{D}$ is a directed poset. A functor is called *finitary* if it preserves filtered (equivalently directed) colimits. An object C is called *finitely presentable* (fp) if the hom-functor $\mathscr{C}(C, -)$ preserves filtered (equivalently directed) colimits, and *finitely generated* (fg) if $\mathscr{C}(C, -)$ preserves directed colimits of monos (i.e. colimits of directed diagrams $D : \mathscr{D} \to \mathscr{C}$

where all connecting morphisms Df are monic in $\mathscr{C}$). Clearly any fp object is fg, but the converse fails in general. In addition, fg objects are closed under strong epis (quotients), which fails for fp objects in general.

A cocomplete category $\mathscr{C}$ is called *locally finitely presentable* (lfp) if there is a set of finitely presentable objects in $\mathscr{C}$ such that every object of $\mathscr{C}$ is a filtered colimit of objects from that set. We refer to [4] for further details.

Examples of lfp categories are the categories of sets, posets and graphs, with finitely presentable objects precisely the finite sets, posets, and graphs, respectively. The category of vector spaces over the field k is lfp with finite-dimensional spaces being the fp-objects. Every algebraic category is lfp. The finitely generated objects are precisely the finitely generated algebras (in the sense of general algebra), and finitely presentable objects are precisely those algebras specified by finitely many generators and finitely many relations.

▶ **Assumptions 4.** For the rest of this section we assume that F denotes a finitary endofunctor on the lfp category $\mathscr{A}$.

The rational fixed point is a fully abstract model of behaviour for all F-coalgebras whose carrier is an fp-object. We now recall its construction [1].

▶ **Notation 5.** Denote by $\mathsf{Coalg}\, F$ the full subcategory of all F-coalgebras on fp carriers, and let $(\varrho F, r)$ be the colimit of the inclusion functor of $\mathsf{Coalg}_{\mathsf{fp}}\, F$ into $\mathsf{Coalg}\, F$: $(\varrho F, r) = \mathrm{colim}(\mathsf{Coalg}_{\mathsf{fp}}\, F \hookrightarrow \mathsf{Coalg}\, F)$ with the colimit injections $a^\sharp : A \to \varrho F$ for every coalgebra $a : A \to FA$ in $\mathsf{Coalg}_{\mathsf{fp}}\, F$.

We call $(\varrho F, r)$ the *rational fixed point* of F; indeed, it is a fixed point:

▶ **Proposition 6** ([1])**.** *The coalgebra structure $r : \varrho F \to F(\varrho F)$ is an isomorphism.*

The rational fixed point can be characterized by a universal property both as a coalgebra and as an algebra for F: as a coalgebra ϱF is the *final locally finitely presentable coalgebra* [17], and as an algebra it is the *initial iterative algebra* [1]. We will not recall the latter notion as it is not needed for the technical development in this paper. Locally finitely presentable (lfp, for short) coalgebras for F can be characterized as precisely those F-coalgebra obtained as a filtered colimit of a diagram of coalgebras from $\mathsf{Coalg}_{\mathsf{fp}}\, F$:

▶ **Proposition 7** ([17], Corollary III.13)**.** *An F-coalgebra is lfp if and only if it is a colimit of some filtered diagram $\mathscr{D} \to \mathsf{Coalg}_{\mathsf{fp}}\, F \hookrightarrow \mathsf{Coalg}\, F$.*

For $\mathscr{A} = \mathsf{Set}$ an F-coalgebra (X, c) is lfp iff it is *locally finite*, i.e. every element of X is contained in a finite subcoalgebra. Analogously, for $\mathscr{A}$ the category of vector spaces over the field k an F-coalgebra (X, c) is lfp iff it is *locally finite dimensional*, i.e. every element of X is contained in a finite dimensional subcoalgebra.

Of course, there is a unique coalgebra morphism $\varrho F \to \nu F$. Moreover, in many cases ϱF is *fully abstract* for lfp coalgebras, i.e. besides being the final lfp coalgebra the above coalgebra morphism is monic; more precisely, if the classes of fp- and fg-objects coincide and F preserves monos, then ϱF is fully abstract (see [7, Proposition 3.12]). The assumption that the two object classes coincide is often true:

▶ **Example 8.**

1. In the category of sets, posets, and graphs, fg-objects are fp and those are precisely the finite sets, posets, and graphs, respectively.
2. A *locally finite variety* is a variety of algebras, where every free algebra on a finite set of generators is finite. It follows that fp- and fg-objects coincide and are precisely the finite

algebras. Concrete examples are the categories of Boolean algebras, distributive lattices and join-semilattices.

3. In the category of $\mathbb{S}$-semimodules for a semiring $\mathbb{S}$ the fp- and fg-objects need not coincide in general. However, if the semiring $\mathbb{S}$ is *Noetherian* in the sense of Ésik and Maletti [11], i.e. every subsemimodule of a finitely generated $\mathbb{S}$-semimodule is itself finitely generated, then fg- and fp-semimodules coincide. Examples of Noetherian semirings are: every finite semiring, every field, every principal ideal domain such as the ring of integers and therefore every finitely generated commutative ring by Hilbert's Basis Theorem. The tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$ is not Noetherian [10]. The usual semiring of natural numbers is also not Noetherian: the $\mathbb{N}$-semimodule $\mathbb{N} \times \mathbb{N}$ is finitely generated but its subsemimodule generated by the infinite set $\{(n, n+1) \mid n \geq 1\}$ is not. However, $\mathbb{N}$-semimodules are precisely the commutative monoids, and for them fg- and fp-objects coincide (this is known as Redei's theorem [23]; see Freyd [13] for a very short proof).
4. Recently, it was established by Sokolova and Woracek [30] that in the category of convex sets, i.e. the Eilenberg-Moore category for the (sub)distribution monad on sets, the classes of fp- and fg-objects coincide.

▶ **Example 9.** We list a number of examples of rational fixed points for cases where they do form subcoalgebras of the final coalgebra.

1. For the functor $FX = \{0,1\} \times X^A$ on Set the finite coalgebras are deterministic automata, and the rational fixed point is carried by the set of regular languages on the alphabet A.
2. For any signature $\Sigma = (\Sigma_n)_{n<\omega}$ of operation symbols with prescribed arity we have the associated polynomial endofunctor on sets given by $F_\Sigma X = \coprod_{n<\omega} \Sigma_n \times X^n$. Its final coalgebra is carried by the set of all (finite and infinite) Σ-trees, i.e. rooted and ordered trees where each node with n-children is labelled by an n-ary operation symbol. The rational fixed point is the subcoalgebra given by rational (or regular [8]) Σ-trees, i.e. those Σ-trees that have only finitely many different subtrees (up to isomorphism) – this characterization is due to Ginali [14]. For example, for the signature Σ with a binary operation symbol $*$ and a constant c the following infinite Σ-tree (here written as an infinite term) is rational:

   $$c * (c * (c * \cdots)));$$

   in fact, its only subtrees are the whole tree and the single node tree labelled by c).
3. For the functor $FX = \mathbb{R} \times X$ on Set the final coalgebra is carried by the set $\mathbb{R}^\omega$ of real streams, and the rational fixed point is carried by its subset of eventually periodic streams (or lassos). Considered as a functor on the category of vector spaces over $\mathbb{R}$, the final coalgebra νF remains the same, but the rational fixed point ϱF consists of all rational streams [26].
4. For the functor $FX = \mathbb{S} \times X^A$ on the category $\mathbb{S}$-Mod of $\mathbb{S}$-semimodules for the semiring $\mathbb{S}$ we already mentioned that $\nu F = \mathbb{S}^{A^*}$ consists of all formal power-series. Whenever the classes of fg- and fp-semimodules coincide, e.g. for every Noetherian semiring $\mathbb{S}$ or the semiring of natural numbers, then ϱF is formed by the *recognizable* formal power-series; from the Kleene-Schützenberger theorem [27] (see also [6]) it follows that these are, equivalently, the *rational* formal power-series.
5. On the category of presheaves $\mathsf{Set}^{\mathcal{F}}$, where $\mathcal{F}$ is the category of all finite sets and maps between them, consider the functor $FX = V + X \times X + \delta(X)$, where $V : \mathcal{F} \hookrightarrow \mathsf{Set}$ is the embedding and $\delta(X)(n) = X(n+1)$. This is a paradigmatic example of a functor arising from a *binding signature* for which initial semantics was studied by Fiore et al. [12].

The final coalgebra νF is carried by the presheaf of all λ-trees modulo α-equivalence: $\nu F(n)$ is the set of (finite and infinite) λ-trees in n free variables (note that such a tree may have infinitely many bound variables). And ϱF is carried by the rational λ-trees, where an α-equivalence class is called *rational* if it contains at least one λ-tree which has (up to isomorphism) only finitely many different subtrees (see [3]). Rational λ-trees also appear as the rational fixed point of a very similar functor on the category of nominal sets [21]. Similarly, for any functor on nominal sets arising from a binding signature [20].

As we mentioned previously, whether fg- and fp-objects coincide is currently unknown in some base categories used in the coalgebraic modelling of systems, for example, in idempotent semirings (used in the treatment of context-free grammars [33]), in algebras for the stack monad (used for modelling configurations of stack machines [15]); or it even fails, for example in the category of finitary monads on sets (used in the categorical study of algebraic trees [2]) or in Eilenberg-Moore categories for a monad in general (the target categories of generalized determinization [28]).

As a remedy, in recent joint work with Pattinson and Wissmann [19], we have introduced the *locally finite fixed point* which provides a fully abstract model of finitely generated behaviour. Its construction is very similar to that of the rational fixed point but based on fg- in lieu of fp-objects. In more detail, one considers the full subcategory $\mathsf{Coalg}_{\mathsf{fg}}\, F$ of all F-coalgebras carried by an fg-object and takes the colimit of its inclusion functor:

$$(\vartheta F, \ell) = \operatorname{colim}(\mathsf{Coalg}_{\mathsf{fg}}\, F \hookrightarrow \mathsf{Coalg}\, F).$$

▶ **Theorem 10** ([19], Theorems 3.10 and 3.12)**.** *Suppose that the finitary functor $F : \mathscr{A} \to \mathscr{A}$ preserves monos. Then $(\vartheta F, \ell)$ is a fixed point for F, and it is a subcoalgebra of νF.*

Furthermore, like its brother, the rational fixed point, ϑF is characterized by a universal property both as a coalgebra and as an algebra: it is the final locally finitely generated coalgebra and the initial fg-iterative algebra [19, Theorems 3.8 and Corollary 3.18].

Under additional assumptions, which all hold in any algebraic category, we have a close relation between ϱF and ϑF; in fact, the following is a consequence of [19, Theorem 3.22]:

▶ **Theorem 11.** *Suppose that $\mathscr{A}$ is an algebraic category and that the finitary functor $F : \mathscr{A} \to \mathscr{A}$ preserves monos. Then ϑF is the image of ϱF in the final coalgebra.*

More precisely, taking the (strong-epi, mono)-factorization of the unique F-coalgebra morphism $\varrho F \to \nu F$ yields ϑF, i.e. for F preserving monos on an algebraic category we have the following picture:

$$\varrho F \twoheadrightarrow \vartheta F \rightarrowtail \nu F.$$

If furthermore, fg- and fp-objects coincide, then $\vartheta F \cong \varrho F$, i.e. the left-hand morphism is an isomorphism.

In the introduction we briefly mentioned a number of interesting instances of ϑF that are not (known to be) instances of the rational fixed point; see [19] for details.

A concrete example, where ϱF is not a subcoalgebra of νF (and hence not isomorphic to ϑF) was given in [7, Example 3.15]. We present a new, simpler example based on similar ideas:

▶ **Example 12.**

1. Let $\mathscr{A}$ be the category of algebras for the signature Σ with two unary operation symbols u und v. The natural numbers $\mathbb{N}$ with the successor function as both operations $u^{\mathbb{N}}$ and $v^{\mathbb{N}}$ form an object of $\mathscr{A}$. We consider the functor $FX = \mathbb{N} \times X$ on $\mathscr{A}$. Coalgebras for F are automata carried by an algebra A in $\mathscr{A}$ equipped with two Σ-algebra morphisms: an output morphism $A \to \mathbb{N}$ and a next state morphism $A \to A$. The final coalgebra is carried by the set $\mathbb{N}^\omega$ of streams of naturals with the coordinatewise algebra operations and with the coalgebra structure given by the usual head and tail functions.
   Note that the free Σ-algebra on a set X of generators is $TX \cong \{u, v\}^* \times X$; we denote its elements by $w(x)$ for $w \in \{u, v\}^*$ and $x \in X$. The operations are given by prefixing words by the letters u and v, respectively: $s^{TX} : w(x) \mapsto sw(x)$ for $s = u$ or v.
   Now one considers the F-coalgebra $a : A \to FA$, where $A = T\{x\}$ is free Σ-algebra on one generator x and a is determined by $a(x) = (0, u(x))$. Clearly, $\dagger a(x)$ is the stream $(0, 1, 2, 3, \cdots)$ of all natural numbers, and since $\dagger a$ is a Σ-algebra morphism we have

   $$\dagger a(u(x)) = \dagger a(v(x)) = (1, 2, 3, 4, \cdots).$$

   Since A is (free) finitely generated, it is of course, finitely presentable as well. Thus, (A, a) is a coalgebra in $\mathsf{Coalg}_{\mathsf{fp}}\, F$. However, one can prove that the (unique) F-coalgebra morphism $a^\sharp : A \to \varrho F$ satisfies $a^\sharp(u(x)) \neq a^\sharp(v(x))$, see the full paper for details [18].
2. In this example we also have that ϑF and νF do not coincide. To see this we use that ϑF is the union of images of all $\dagger a : TX \to \nu F$ where (TX, a) ranges over those F-coalgebras whose carrier TX is free finitely generated (i.e. TX is a term algebra over some finite set X) [19, Theorem 4.4].
   Note that being a Σ-algebra morphism any coalgebra structure $a : TX \to FTX$ is determined by its action on the generators. And from the form of any TX we know that for any $x \in X$ there exist $k, n_i \in \mathbb{N}$, $w_i \in \{u, v\}^*$ and $x_i \in X$, $i = 1, \ldots k$, such that $x = x_0$ and

   $$\begin{aligned} a(x_i) &= (n_i, w_i(x_{i+1})) && \text{for } i = 0, \ldots, k-1 \text{ and} \\ a(x_k) &= (n_k, w_k(x_j)) && \text{for some } j \in \{0, \ldots, k\}. \end{aligned}$$

   Now let $m_i = |w_i|$, $i = 1, \ldots, k$, be the lengths of words. Then it follows that

   $$\dagger a(x_0) = (n_0, m_0 + n_1, m_0 + m_1 + n_2, \cdots, m_0 + \cdots + m_{k-1} + n_k, m_0 + \cdots m_k + n_j, \cdots).$$

   Let m be the maximum of all n_i and m_i. Then it is clear that the n-th entry of $\dagger a(x_0)$ can be at most $(n+1) \cdot m$. It follows that for any $w \in \{u, v\}^*$ the n-th entry of $\dagger a(w(x))$ is bounded above by $(n+1) \cdot m + |w|$. Thus, the entries of every stream in ϑF grow at most linearly. However, there are streams in νF for which this is not the case, e.g. the stream $(1, 2, 4, 8, \cdots)$ of powers of 2. Hence ϑF does not coincide with νF.

3 Proper Functors and Coalgebras Carried by Free Algebras

The purpose of this section is to study the situation where the rational fixed point for a functor F on an algebraic category Set^T coincides with the locally finite one, and moreover, both can be constructed just from those coalgebras whose carrier is a free finitely generated coalgebra. The latter coalgebras are precisely those coalgebras arising as the results of the generalized determinization [28].

▶ **Assumptions 13.** Throughout the rest of the paper we assume that $\mathscr{A}$ is an *algebraic category*, i.e. $\mathscr{A}$ is (equivalent to) the Eilenberg-Moore category Set^T for a finitary monad T on Set. Furthermore, we assume that $F : \mathscr{A} \to \mathscr{A}$ is a finitary endofunctor preserving surjective T-algebra morphisms.

▶ Remark 14.

1. The most common instance is when F is a lifting of an endofunctor $F_0 : \mathsf{Set} \to \mathsf{Set}$, i.e. we have a commutative square $F_0 \cdot U = U \cdot F$, where $U : \mathscr{A} \to \mathsf{Set}$ is the forgetful functor. Then F preserves surjective T-algebra morphisms since every set functor F_0 preserves surjections (which are split epis in Set). In addition, F is finitary whenever F_0 is so because filtered colimits in Set^T are created by U. Furthermore, observe that the assumption that F preserves monomorphisms in Theorems 10 and 11 as well as in Corollary 28 is not needed. Indeed, inspection of the proofs in [19] reveals that it suffices to assume that non-empty monomorphisms are preserved, and this holds for every lifted F since it does for every F_0 on Set.
2. Let $F : \mathsf{Set} \to \mathsf{Set}$ have a lifting to Set^T (also denoted by F for simplicity). *Generalized determinization* [28] is the process of turning a given coalgebra $c : X \to FTX$ in Set into the coalgebra $c^\star : TX \to FTX$ for the lifting of F on Set^T. For example, for the functor $FX = \{0,1\} \times X^\Sigma$ on Set and the finite power-set monad $T = \mathcal{P}_\mathsf{f}$, FT-coalgebras are precisely non-deterministic automata and generalized determinization is the construction of a deterministic automaton by the well-known subset construction. The unique F-coalgebra morphism $\dagger(c^\star)$ assigns to each state $x \in X$ the language accepted by x in the given nondeterministic automaton (whereas the final semantics for FT on Set provides a kind of process semantics taking the nondeterministic branching into account).
   Thus studying the behaviour of F-coalgebras whose carrier is a free finitely generated T-algebra TX is precisely the study of a *coalgebraic language semantics* of finite FT-coalgebras.

▶ **Notation 15.** We denote by $\mathsf{Coalg}_\mathsf{free} F$ the full subcategory of $\mathsf{Coalg}\, F$ given by all coalgebras $c : TX \to FTX$ whose carrier is a free finitely generated T-algebra, i.e. where X is a finite set X.

The colimit of the inclusion functor of $\mathsf{Coalg}_\mathsf{free} F$ into the category of all F-coalgebras is denoted by $(\varphi F, \zeta) = \mathrm{colim}(\mathsf{Coalg}_\mathsf{free} F \hookrightarrow \mathsf{Coalg}\, F)$ with the colimit injections $\mathsf{in}_c : TX \to \varphi F$ for every $c : TX \to FTX$.

▶ **Notation 16.** Since $\mathsf{Coalg}_\mathsf{free} F$ is a full subcategory of $\mathsf{Coalg}_\mathsf{fp} F$, the universal property of the colimit φF induces a coalgebra morphism denoted by $h : \varphi F \to \varrho F$. Furthermore we write $m : \varphi F \to \nu F$ for the unique F-coalgebra morphisms into the final lfp coalgebra and the final coalgebra, respectively.

▶ Remark 17. We shall show in Propositon 21 that h is a strong epimorphism. Thus, whenever F preserves monos, we have the picture (1) from the introduction.

Urbat [31] shows that φF is always a fixed point of F. However, φF does not have a universal property similar to the coalgebras ϱF and ϑF. In fact, Urbat gives the following example of a coalgebra $c : TX \to FTX$ where $\mathsf{in}_c : TX \to \varphi F$ is not the only F-coalgebra morphism:

▶ **Example 18.**

1. Let $\mathscr{A}$ be the category of algebras for the signature with one unary operation symbol u (and no equations), and let $F = \mathsf{Id}$ be the identity functor on $\mathscr{A}$. Let A be the free (term) algebra on one generator x, and let B be the free algebra on one generator y

(i.e. both A and B are isomorphic to $\mathbb{N}$). We equip A and B with the F-coalgebra structures $a = \mathsf{id} : A \to A$ and $b : B \to B$ given by $b(y) = u(y)$. Define $g : A \to \varphi F$ by $g(x) = \mathsf{in}_b(y)$. Then one can show that g is an F-coalgebra morphism different from the F-coalgebra morphism $\mathsf{in}_a : A \to \varphi F$.

2. Using similar ideas as in the previous point one can show that, for the category $\mathscr{A}$ and $FX = \mathbb{N} \times X$ from Example 12, φF and ϱF do not coincide, see the full paper [18]. Consequently, in this example, none of the arrows in (1) is an isomorphism.

In this section we are going to investigate when the first three fixed points in (1) collaps to one, i.e. $\varphi F \cong \varrho F \cong \vartheta F$. As a consequence, it follows that finality of a given lfp coalgebra for F can be established by checking the universal property only for the coalgebras in $\mathsf{Coalg}_{\mathsf{free}}\, F$ (Corollary 29).

▶ **Lemma 19.** *The category* $\mathsf{Coalg}_{\mathsf{free}}\, F$ *is closed under finite coproducts.*

Proof. The empty map $0 \to FT0$ extends uniquely to a T-algebra morphism $T0 \to FT0$, i.e. an F-coalgebra, and this coalgebra is the initial object of $\mathsf{Coalg}_{\mathsf{free}}\, F$.

Given coalgebras $c : TX \to FTX$ and $d : TY \to FTY$ one uses that $T(X + Y)$ together with the injections $T\mathsf{inl} : TX \to T(X + Y)$ and $T\mathsf{inr} : TY \to T(X + Y)$ form a coproduct in Set^T. This implies that forming the coproduct of (TX, c) and (TY, d) in $\mathsf{Coalg}\, F$ we obtain an F-coalgebra on $T(X + Y)$, and this is an object of $\mathsf{Coalg}_{\mathsf{free}}\, F$ since $X + Y$ is finite. ◀

▶ Remark 20. We will use later that the colimit φF is a sifted colimit.

1. Recall that a small category $\mathscr{D}$ is called *sifted* [5] if finite products commute with colimits over $\mathscr{D}$ in Set. More precisely, $\mathscr{D}$ is sifted iff given any diagram $D : \mathscr{D} \times \mathscr{J} \to \mathsf{Set}$, where $\mathscr{J}$ is a finite discrete category, the canonical map

$$\operatorname*{colim}_{d \in \mathscr{D}} \Big(\prod_{j \in \mathscr{J}} D(d, j) \Big) \to \prod_{j \in \mathscr{J}} (\operatorname*{colim}_{d \in \mathscr{D}} D(d, j))$$

is an isomorphism. A *sifted colimit* is a colimit of a diagram with a sifted diagram scheme.

2. It is well-known that the forgetful functor $\mathsf{Set}^T \to \mathsf{Set}$ preserves sifted colimits; this follows from [5, Proposition 2.5].
3. Further recall [5, Example 2.16] that every small category $\mathscr{D}$ with finite coproducts is sifted. Thus, following Lemma 19, $\mathscr{D} = \mathsf{Coalg}_{\mathsf{free}}\, F$ is sifted, and φF is a sifted colimit.

▶ **Proposition 21.** *The above morphism* $h : \varphi F \twoheadrightarrow \varrho F$ *is a strong epimorphism in* $\mathscr{A}$*.*

▶ Remark 22.

1. Recall that a *zig-zag* in a category $\mathscr{A}$ is a diagram of the form

$$Z_0 \xrightarrow{f_0} Z_1 \xleftarrow{f_1} Z_2 \xrightarrow{f_2} Z_3 \xleftarrow{f_3} \cdots \xrightarrow{f_{n-2}} Z_{n-1} \xleftarrow{f_{n-1}} Z_n.$$

For $\mathscr{A} = \mathsf{Set}^T$, we say that the zig-zag *relates* $z_0 \in Z_0$ and $z_n \in Z_n$ if there exist $z_i \in Z_i$, $i = 1, \ldots, n-1$ such that $f_i(z_i) = z_{i+1}$ for i even and $f_i(z_{i+1}) = z_i$ for i odd.

2. Ésik and Maletti [10] introduced the notion of a *proper* semiring in order to obtain the decidability of the (language) equivalence of weighted automata. A semiring $\mathbb{S}$ is called *proper* if, whenever we have two $\mathbb{S}$-weighted automata A and B and two states x in A and y in B that accept the same weighted language, then there exists a zig-zag

$$A = M_0 \to M_1 \leftarrow M_2 \to M_3 \leftarrow \cdots \to M_{n-1} \leftarrow M_n = B$$

of simulations that *relates* x and y. They show that every Noetherian semiring is proper as well as the semiring $\mathbb{N}$ of natural numbers, which is not Noetherian. However, the tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$ is not proper.

Recall from Example 2 that $\mathbb{S}$-weighted automata with input alphabet Σ are equivalently coalgebras with carrier $\mathbb{S}^n$, for some $n \geq 1$, for the functor $FX = \mathbb{S} \times X^\Sigma$ on the category $\mathbb{S}$-Mod. Thus, since simulations are precisely the F-coalgebra morphisms, one easily generalizes the notion of a proper semiring as follows. Recall that $\eta_X : X \to TX$ denotes the unit of the monad T.

▶ **Definition 23.** We call the functor $F : \mathscr{A} \to \mathscr{A}$ *proper* whenever for every pair of coalgebras $c : TX \to FTX$ and $d : TY \to FTY$ in $\mathsf{Coalg}_{\mathsf{free}}\, F$ and every $x \in X$ and $y \in Y$ such that $\eta_X(x) \sim \eta_Y(y)$ are behaviourally equivalent there exists a zig-zag in $\mathsf{Coalg}_{\mathsf{free}}\, F$ relating $\eta_X(x)$ and $\eta_Y(y)$.

▶ **Example 24.** A semiring $\mathbb{S}$ is proper iff the functor $FX = \mathbb{S} \times X^\Sigma$ on $\mathbb{S}$-Mod is proper.

▶ **Example 25.** Constant functors are always proper. Indeed, suppose that F is the constant functor on some algebra A. Then we have $\nu F = A$, and for any F-coalgebra B its coalgebra structure $c : B \to FB = A$ is also the unique F-coalgebra morphism from B to $\nu F = A$.

Now given any $c : TX \to FTX = A$ and $d : TY \to FTY = A$ and $x \in TX$, $y \in TY$ as in Definiton 23. Then $\eta_X(x) \sim \eta_Y(y)$ is equivalent to $c(\eta_X(x)) = d(\eta_Y(y))$. Let a be this element of A, and extend $x : 1 \to X$, $y : 1 \to Y$ and $a : 1 \to A$ to T-algebra morphisms $x^\star : T1 \to TX$, $y^\star : T1 \to TY$ and $a^\star : T1 \to A = FT1$ (the latter yielding an F-coalgebra). Then $TX \xleftarrow{x^\star} T1 \xrightarrow{y^\star} TY$ is the required zig-zag in $\mathsf{Coalg}_{\mathsf{free}}\, F$ relating $\eta_X(x)$ and $\eta_Y(y)$.

In general, it seems to be non-trivial to establish that a given functor is proper (even for the identity functor this may fail as we have seen in Example 18.1). However, we will provide in Proposition 30 sufficient conditions on $\mathscr{A}$ and F the entail properness using our main result:

▶ **Theorem 26.** *The functor F is proper iff the coalgebra φF is a subcoalgebra of νF.*

The latter condition states that the unique coalgebra morphism $m : \varphi F \to \nu F$ is a monomorphism in $\mathscr{A}$.

We present the proof of this theorem Section 4. Here we continue with a discussion of the consequences of this result.

▶ **Corollary 27.** *If F is proper, then φF is the rational fixed point of F.*

Proof. Let $u : \varrho F \to \nu F$ be the unique F-coalgebra morphism. Then we have a commutative triangle of F-coalgebra morphisms due to finality of νF: $m = (\varphi F \overset{h}{\twoheadrightarrow} \varrho F \overset{u}{\to} \nu F)$. Since F is proper m is a monomorphism in $\mathscr{A}$, hence so is h. Since h is also a strong epimorphism by Proposition 21, it is an isomorphism. Thus, $\varphi F \cong \varrho F$ is the rational fixed point of F. ◀

▶ **Corollary 28.** *If finitely generated and finitely presentable algebras coincide in $\mathscr{A}$ and F preserves monos, then F is proper iff $\varphi F \cong \varrho F \cong \vartheta F \rightarrowtail \nu F$.*

Indeed, this follows from Corollary 27 and Theorem 11. Note that this also entails full abstractness of $\varphi F \cong \varrho F$.

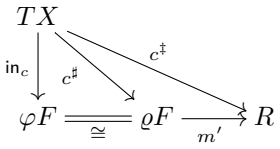
A key result for establishing soundness and completeness of coalgebraic regular expression calculi is the following corollary (cf. [7, Corollary 3.36] and its applications in Sections 4 and 5 of loc. cit.).

▶ **Corollary 29.** *Suppose that F is proper. Then an F-coalgebra (R, r) is a final lfp coalgebra if and only if (R, r) is lfp and for every coalgebra (TX, c) in $\mathsf{Coalg}_{\mathsf{free}}\, F$ there exists a unique F-coalgebra morphism from TX to R.*

Proof. The implication "⇒" clearly holds

For "⇐" it suffices to prove that for every $a : A \to FA$ in $\mathsf{Coalg}_{\mathsf{fp}}\, F$ there exists a unique F-coalgebra morphism from A to R. In fact, it then follows that R is the final lfp coalgebra. To see this write an arbitrary lfp coalgebra A as a filtered colimit of a diagram $D : \mathscr{D} \to \mathsf{Coalg}_{\mathsf{fp}}\, F \hookrightarrow \mathsf{Coalg}\, F$ with colimit injections $h_d : Dd \to A$ (d an object in $\mathscr{D}$). Then the unique F-coalgebra morphisms $u_d : Dd \to R$ form a compatible cocone, and so one obtains a unique $u : A \to R$ such that $u \cdot h_d = u_d$ holds for every object d of $\mathscr{D}$. It is now straightforward to prove that u is a unique F-coalgebra morphism from A to R.

Now let $a : A \to FA$ be a coalgebra in $\mathsf{Coalg}_{\mathsf{fp}}\, F$. For every (TX, c) in $\mathsf{Coalg}_{\mathsf{free}}\, F$ denote by $c^\ddagger : TX \to R$ the unique F-coalgebra morphism that exists by assumption. These morphisms $c^\ddagger$ form a compatible cocone of the diagram $\mathsf{Coalg}_{\mathsf{free}}\, F \hookrightarrow \mathsf{Coalg}\, F$. Thus, we obtain a unique F-coalgebra morphism $m' : \varphi F \cong \varrho F \to R$ such that the following diagram commutes for every $c : TX \to FTX$ in $\mathsf{Coalg}_{\mathsf{free}}\, F$:



Therefore we have an F-coalgebra morphism

$$h = (A \xrightarrow{a^\sharp} \varrho F \xrightarrow{m'} R).$$

To prove it is unique, assume that $g : A \to R$ is any F-coalgebra morphism. As in the proof of Proposition 21, we know that A is the quotient of some TX in $\mathsf{Coalg}_{\mathsf{free}}\, F$ via $q : TX \twoheadrightarrow A$, say. Then we have $m' \cdot a^\sharp \cdot q = g \cdot q$ because there is only one F-coalgebra morphism from TX to R by hypothesis. It follows that $h = m' \cdot a^\sharp = g$ since q is epimorphic. ◀

The next result provides sufficient conditions for properness of F. It can be seen as a category-theoretic generalization of Ésik's and Maletti's result [10, Theorem 4.2] that Noetherian semirings are proper.

For the special case of a lifting F of a set functor F_0 this is a corollary of a result of Winter [32, Proposition 7].

▶ **Proposition 30.** *Suppose that finitely generated algebras in $\mathscr{A}$ are closed under kernel pairs and that F maps kernel pairs to weak pullbacks in* Set*. Then F is proper.*

Note that closure of finitely generated algebras under kernel pairs can equivalently be stated in general algebra terms as follows: every congruence R of a finitely generated algebra A is finitely generated as a subalgebra $R \hookrightarrow A \times A$ (observe that this is *not* equivalent to stating that R is a finitely generated congruence).

Furthermore, for a lifting F of a set functor F_0, the above condition on F holds whenever F preserves weak pullbacks. Hence, all the functors on algebraic categories mentioned in Example 9 satisfy this assumption.

▶ **Examples 31.**

1. The first condition in Proposition 30 is not necessary for properness of F. In fact, it fails in the category of semimodules for $\mathbb{N}$, viz. the category of commutative monoids: the submonoid of $\mathbb{N}\times\mathbb{N}$ infinitely generated by $\{(n, n+1) \mid n \in \mathbb{N}\}$ is not a finitely generated submonoid. However, as we mentioned in Example 24, $FX = \mathbb{N} \times X^\Sigma$ is proper on the category of commutative monoids.
2. In Example 8.4 we mentioned that, in the category of convex sets (i.e. Eilenberg-Moore algebras for the distribution monad), fg- and fp-objects coincide. However, fg-objects are not closed under kernel pairs. In fact, the interval $[0, 1]$ is the free convex set on two generators, but $\{(0, 0), (1, 1)\} \cup (0, 1) \times (0, 1)$ is a congruence on $[0, 1]$ that is not an fg-object (i.e. a polytope) [30, Example 4.13]. It is an open problem whether coalgebraic type functors of interest on convex sets are proper, e.g. the functor $FX = [0, 1] \times X^\Sigma$.

4 Proof of Theorem 26

In this section we will present the proof of our main technical result Theorem 26. We start with two technical lemmas.

▶ Remark 32. Recall [5, Proposition 11.28.2] that every free T-algebra TX is *perfectly presentable*, i.e. the hom-functor $\mathsf{Set}^T(TX, -)$ preserves sifted colimits. It follows that for every sifted diagram $D : \mathscr{D} \to \mathsf{Set}^T$ and every T-algebra morphism $h : TX \to \operatorname{colim} D$ there exists some $d \in \mathscr{D}$ and $h' : TX \to Dd$ such that $h = \mathsf{in}_d \cdot h'$.

▶ **Lemma 33.** *For every finite set X and map $f : X \to \varphi F$ there exists an object (TY, d) in $\mathsf{Coalg}_{\mathsf{free}}\, F$ and a map $g : X \to Y$ such that $f = (X \xrightarrow{g} Y \xrightarrow{\eta_Y} TY \xrightarrow{\mathsf{in}_d} \varphi F)$.*

▶ Remark 34. Recall that a colimit of a diagram $D : \mathscr{D} \to \mathsf{Set}$ is computed as follows:

$$\operatorname{colim} D = \Big(\coprod_{d\in\mathscr{D}} Dd\Big)/{\sim},$$

where $\sim$ is the least equivalence on the coproduct (i.e. the disjoint union) of all Dd with $x \sim Df(x)$ for every $f : d \to d'$ in $\mathscr{D}$ and every $x \in Dd$. In other words, for every pair of objects c, d of $\mathscr{D}$ and $x \in Dc$, $y \in Dd$ we have $x \sim y$ iff there is a zig-zag in $\mathscr{D}$ whose D-image relates x and y (cf. Remark 22).

▶ **Lemma 35.** *Let (TX, c) and (TY, d) be coalgebras in $\mathsf{Coalg}_{\mathsf{free}}\, F$, $x \in TX$, and $y \in TY$. Then the following are equivalent:*

1. $\mathsf{in}_c(x) = \mathsf{in}_d(y) \in \varphi F$, *and*
2. *there is a zig-zag in $\mathsf{Coalg}_{\mathsf{free}}\, F$ relating x and y.*

Proof. By Remark 20, φF is a sifted colimit. Hence, the forgetful functor $\mathsf{Coalg}\, F \to \mathsf{Set}^T \to \mathsf{Set}$ preserves this colimit. Thus the colimit φF is formed as recalled in Remark 34:

$$\varphi F \cong \Big(\coprod_c TX_c\Big)/{\sim},$$

where $c : TX_c \to FTX_c$ ranges over the objects of $\mathsf{Coalg}_{\mathsf{free}}\, F$. Therefore, we have the desired equivalence. ◀

Proof of Theorem 26. "⇒" Suppose that for $m : \varphi F \to \nu F$ we have $x, y \in \varphi F$ with $m(x) = m(y)$. We apply Lemma 33 to $1 \xrightarrow{x} \varphi F$ and $1 \xrightarrow{y} \varphi F$, respectively, to obtain two objects $c : TX \to FTX$ and $d : TY \to FTY$ in $\mathsf{Coalg}_{\mathsf{free}}\, F$ with $x' \in X$ and $y' \in Y$ such

that $\mathsf{in}_c(\eta_X(x')) = x$ and $\mathsf{in}_d(\eta_Y(y')) = y$. By the uniqueness of coalgebra morphisms into νF we have

$$\dagger c = m \cdot \mathsf{in}_c \qquad \text{and} \qquad \dagger d = m \cdot \mathsf{in}_d. \tag{2}$$

Thus we compute:

$$\dagger c(\eta_X(x')) = m \cdot \mathsf{in}_c \cdot \eta_X(x') = m(x) = m(y) = m \cdot \mathsf{in}_d \cdot \eta_Y(y') = \dagger d(\eta_Y(y')).$$

Since F is proper by assumption, we obtain a zig-zag in $\mathsf{Coalg}_{\mathsf{free}}\, F$ relating $\eta_X(x')$ and $\eta_Y(y')$. Thus, these two elements are merged by the colimit injections, and we have $x = \mathsf{in}_c(\eta_X(x')) = \mathsf{in}_d(\eta_Y(y') = y$. We conclude that m is monomorphic.

"$\Leftarrow$" Suppose that $m : \varphi F \rightarrowtail \nu F$ is a monomorphism. Let $c : TX \to FTX$ and $d : TY \to FTY$ be objects of $\mathsf{Coalg}_{\mathsf{free}}\, F$, and let $x \in X$ and $y \in Y$ be such that $\dagger c(\eta_X(x)) = \dagger d(\eta_Y(y))$. Using (2) and the fact that m is monomorphic we get $\mathsf{in}_c(\eta_X(x)) = \mathsf{in}_d(\eta_Y(y))$. By Lemma 35, we thus obtain a zig-zag in $\mathsf{Coalg}_{\mathsf{free}}\, F$ relating $\eta_X(x)$ and $\eta_Y(y)$. This proves that F is proper. ◀

5 Conclusions and Further Work

Inspired by Ésik and Maletti's notion of a proper semiring, we have introduced the notion of a proper functor. We have shown that, for a proper endofunctor F on an algebraic category preserving regular epis and monos, the rational fixed point ϱF is fully abstract and moreover determined by those coalgebras with a free finitely generated carrier (i.e. the target coalgebras of generalized determinization).

Our main result also shows that properness is necessary for this kind of full abstractness. For categories in which fg-objects are closed under kernel pairs we saw that when F maps kernel pairs to weak pullbacks in Set, then it is proper. This provides a number of examples of proper functors. However, in several categories of interest the condition on kernel pairs fails, e.g. in $\mathbb{N}$-semimodules (commutative monoids) and convex sets. There can still be proper functors, e.g. $FX = \mathbb{N} \times X^\Sigma$ on the former. But establishing properness of a functor without using Proposition 30 seems non-trivial, and we leave this task as an open problem for further work.

One immediate consequence of our results is that the soundness and completeness of the expression calculi for weighted automata [7] extend from Noetherian to proper semirings, see Ésik and Kuich [9] for a related result.

In the future, when additional proper functors are known, it will be interesting to study regular expression calculi for their coalgebras and use the technical machinery developed in the present paper for soundness and completeness proofs.

References

1 Jiří Adámek, Stefan Milius, and Jiří Velebil. Iterative algebras at work. *Math. Structures Comput. Sci.*, 16(6):1085–1131, 2006.

2 Jiří Adámek, Stefan Milius, and Jiří Velebil. On second-order iterative monads. *Theoret. Comput. Sci.*, 412:4969–4988, 2011.

3 Jiří Adámek, Stefan Milius, and Jiří Velebil. Semantics of higher-order recursion schemes. *Log. Methods Comput. Sci.*, 7(1:15):43 pp., 2011.

4 Jiří Adámek and Jiří Rosický. *Locally presentable and accessible categories.* Cambridge University Press, 1994.

5 Jiří Adámek, Jiří Rosický, and Enrico Vitale. *Algebraic Theories*. Cambridge University Press, 2011.

6 Jean Berstel and Christophe Reutenauer. *Rational Series and Their Languages*. Springer-Verlag, 1988.

7 Marcello Bonsangue, Stefan Milius, and Alexandra Silva. Sound and complete axiomatizations of coalgebraic language equivalence. *ACM Trans. Comput. Log.*, 14(1:7), 2013.

8 Bruno Courcelle. Fundamental properties of infinite trees. *Theoret. Comput. Sci.*, 25:95–169, 1983.

9 Zoltán Ésik and Werner Kuich. Free iterative and iteration k-semialgebras. *Algebra Univ.*, 67(2):141–162, 2012.

10 Zoltán Ésik and Andreas Maletti. Simulation vs. equivalence. In *Proc. 6th Int. Conf. Foundations of Computer Science*, pages 119–122. CSREA Press, 2010.

11 Zoltán Ésik and Andreas Maletti. Simulations of weighted tree automata. In *Proc. CIAA'11*, volume 6482 of *Lecture Notes Comput. Sci.*, pages 321–330. Springer, 2011.

12 Marcelo Fiore, Gordon D. Plotkin, and Daniele Turi. Abstract syntax and variable binding. In *Proc. LICS'99*, pages 193–202. IEEE Press, 1999.

13 Peter Freyd. Rédei's finiteness theorem for commutative semigroups. *Proc. Amer. Math. Soc.*, 19(4), 1968.

14 Susanna Ginali. Regular trees and the free iterative theory. *J. Comput. System Sci.*, 18:228–242, 1979.

15 Sergey Goncharov, Stefan Milius, and Alexandra Silva. Towards a coalgebraic Chomsky hierarchy (extended abstract). In *Proc. TCS'14*, volume 8705 of *Lecture Notes Comput. Sci.*, pages 265–280. Springer, 2014.

16 Bart Jacobs, Alexandra Silva, and Ana Sokolova. Trace semantics via determinization. *J. Comput. System Sci.*, 2015.

17 Stefan Milius. A sound and complete calculus for finite stream circuits. In *Proc. LICS'10*, pages 449–458. IEEE Computer Society, 2010.

18 Stefan Milius. Proper functors and their rational fixed point. Full version; available at `https://arxiv.org/abs/1705.09198`, 2017.

19 Stefan Milius, Dirk Pattinson, and Thorsten Wißmann. A new foundation for finitary corecursion: The locally finite fixpoint and its properties. In *Proc. FoSSaCS'16*, volume 9634 of *Lecture Notes Comput. Sci. (ARCoSS)*, pages 107–125. Springer, 2016.

20 Stefan Milius, Lutz Schröder, and Thorsten Wißmann. Regular behaviours with names: On rational fixpoints of endofunctors on nominal sets. *Appl. Categ. Structures*, 24(5):663–701, 2016.

21 Stefan Milius and Thorsten Wißmann. Finitary corecursion for the infinitary lambda calculus. In *Proc. CALCO'15*, volume 35 of *LIPIcs*, pages 336–351, 2015.

22 Ion Petre and Arto Salomaa. Algebraic systems and pushdown automata. In *Handbook of Weighted Automata*, pages 257–289. Springer, 2009.

23 Lásló Rédei. *The Theory of Finitely Generated Commutative Semigroups*. Pergamon, Oxford-Edinburgh-New York, 1965.

24 Jan Rutten. Automata and coinduction (an exercise in coalgebra). In *Proc. CONCUR 1998*, volume 1466 of *Lecture Notes Comput. Sci.*, pages 194–218. Springer, 1998.

25 Jan Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249(1):3–80, 2000.

26 Jan Rutten. Rational streams coalgebraically. *Log. Methods Comput. Sci.*, 4(3:9):22 pp., 2008.

27 Marcel Paul Schützenberger. On the definition of a family of automata. *Inform. and Control*, 4(2–3):275–270, 1961.

28 Alexandra Silva, Filippo Bonchi, Marcello Bonsangue, and Jan Rutten. Generalizing determinization from automata to coalgebras. *Log. Methods Comput. Sci*, 9(1:9), 2013.

29 Alexandra Silva, Marcello Bonsangue, and Jan Rutten. Non-deterministic Kleene coalgebras. *Log. Methods Comput. Sci.*, 6(3:23):39 pp., 2010.

30 Ana Sokolova and Harald Woracek. Congruences of convex algebras. *J. Pure Appl. Algebra*, 219(8):3110–3148, 2015.

31 Henning Urbat. Finite behaviours and finitary corecursion. In *Proc. CALCO'17*, volume 72 of *LIPIcs*, pages 24:1–24:15, 2017.

32 Joost Winter. A completeness result for finite λ-bisimulations. In *Proc. FoSSaCS'15*, volume 9034 of *Lecture Notes Comput. Sci.*, pages 117–132, 2015.

33 Joost Winter, Marcello Bonsangue, and Jan Rutten. Coalgebraic characterizations of context-free languages. *Log. Methods Comput. Sci.*, 9(3), September 2013.

34 Joost Winter, Marcello Bonsangue, and Jan Rutten. Context-free coalgebras. *J. Comput. System Sci.*, 81(5):911–939, 2015.

A Classical Groupoid Model for Quantum Networks*

David Reutter[1] and Jamie Vicary[2]

1 **Department of Computer Science, University of Oxford, UK**
`david.reutter@cs.ox.ac.uk`
2 **Department of Computer Science, University of Oxford, UK**
`jamie.vicary@cs.ox.ac.uk`

Abstract

We give a mathematical analysis of a new type of classical computer network architecture, intended as a model of a new technology that has recently been proposed in industry. Our approach is based on groubits, generalizations of classical bits based on groupoids. This network architecture allows the direct execution of a number of protocols that are usually associated with quantum networks, including teleportation, dense coding and secure key distribution.

1998 ACM Subject Classification C.2.1 Network Architecture and Design, F.4.3 Formal Languages

Keywords and phrases groupoids, networks, quantum, semantics, key distribution

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.19

1 Introduction

Borrill and Karp have recently introduced the notion of *timeless network* [9], a new paradigm for distributed communication currently under commercial development by Earth Computing[1]. Inspired by their proposal, we introduce a new network architecture based on *groubits*—group-theoretical generalizations of classical bits, with similar behaviour to qubits in quantum information—and go on to show that groubits can be manipulated to achieve a wide range of surprising informatic tasks. We give a categorical syntax and semantics for groubits, and develop a graphical calculus to prove correctness of groubit protocols.

Groubits. A groubit is a computational device storing two ordinary bits (A_L, A_I), a *logical bit* A_L and an *internal bit* A_I, and supporting the primitive operations **Init**, **Swap**, **Read**, **Write** and **Tick**. Some of these operations in turn make use of the procedure **Rand**, a function with no arguments which returns either 0 or 1 nondeterministically. We describe these procedures as follows, in their simplest instantiations. The **Init** operation takes no arguments, and creates a new groubit in the following state:

- $\mathbf{Init} = (\mathbf{Rand}, 0)$

Here and throughout, we intend that the **Rand** function is executed freshly each time. The **Swap** operation acts on a groubit, exchanging the logical and internal bits:

- $\mathbf{Swap}(A_L, A_I) = (A_I, A_L)$

Conventional single bits $[B]$ can be stored in groubits, using the following read and write procedures:

* An extended version of this paper can be found at [28], `https://arxiv.org/abs/1707.00966`.
[1] See `http://www.earthcomputing.io`.

- $\mathbf{Read}(A_L, A_I) = [A_L]$
- $\mathbf{Write}[B] = (B, \mathbf{Rand})$

The **Read** operation destroys a groubit and creates a conventional bit, while the **Write** operation destroys a conventional bit and creates a new groubit. Pairs of groubits can also be connected by a *link*, enabling the **Tick** operation, where A and B label the two connected nodes, and $\oplus$ is addition modulo 2:

- $\mathbf{Tick}((A_L,A_I),(B_L,B_I)) = ((A_L,A_I \oplus B_L),(B_L,B_I \oplus A_L))$

Intuitively, for each node in the pair, we flip the internal bit just when the other node has logical bit equal to 1. Nodes can belong to multiple links, forming a graph topology.

Assumptions. We make some assumptions about these groubit operations.

- **Atomicity.** The operations **Init**, **Swap**, **Read**, **Write** and **Tick** are atomic.
- **Security.** The state of a node cannot be accessed, except via **Read**.

We emphasize that claims we make about the functionality of groubit networks—in particular, security properties—rest on the validity of these assumptions.[2] We suggest that these assumptions are within the realm of technological plausibility; for example, separation kernels [34] are a well-developed technology for guaranteeing strong security properties of private memory states within embedded devices. Our focus here is on the logical properties of these devices, rather than on questions of implementation, so we do not discuss these aspects further. Note however that we do not assume that devices cannot fail; to satisfy the assumptions, it would be valid for a device to self-destruct if tampering was detected.

1.1 Significance

We claim that groubits have exotic properties making them interesting to study. In particular, they allow timeout-free atomic message routing (the origin of the term 'timeless network'), and they have the ability to replicate a variety of quantum protocols.

Message routing. Linear chains of groubits allow message routing between nodes with guaranteed message atomicity, and without timeouts (see Section 3.1). We understand that developing this idea is the primary commercial interest of Earth Computing, with a focus on high-resilience network architectures for data centres; this is potentially significant, since the timeout properties of the standard TCP transport protocol [16] can cause reliability issues in a data centre environment [1, 9].

Quantum behaviour. A range of quantum protocols—entanglement creation, teleportation, dense coding, and secure key distribution—can be implemented on a groubit network, almost without modification.

If groubit networks can be implemented at scale in the real world, this may prove technologically significant, given the possibility that quantum computers may within decades be able to break in polynomial time the RSA public-key encryption scheme which is currently technologically dominant [8]. Should this possibility be realized, it has been suggested that quantum key distribution could be used as an alternative technology to enable long-range information theoretically–secure communication [15]; we suggest that key distribution running on a large-scale groubit network may be an alternative worth investigating.

[2] For quantum protocols such as quantum key distribution, security is derivable from the laws of physics; this is not the case here [32].

Some points must be made completely clear. Information theoretically–secure key distribution is known to be impossible in a classical computation setting. Our claim that it can be implemented using networks of groubits rests on the atomicity and security assumptions given in Section 1, and will hold for any real-world implementation only to some approximation. Also, we do not claim that *all* quantum protocols or algorithms can be implemented on groubits; in particular, we expect no analogue of 'quantum speedup', and give no classical model for important procedures such as the Grover or Shor algorithms [24].

Nonetheless, for those quantum protocols that we claim can operate on a groubit network, we mean this in a strong sense. In an extended version of this paper [28], we present a quantization functor which gives a structure-preserving mapping from our setting into quantum theory, sending groubit protocols to quantum protocols, and sending a groubit to a Hadamard matrix [27,33]. In other words, groubits yield a local hidden variable model for the part of quantum theory in the image of this quantization functor.

1.2 Overview

The structure of this article is as follows. In Section 2, we give the definition of a groubit in terms of groupoids with extra structure. We define the 2-category $\mathbf{GpdAct_s}$ of finite groupoids, free profunctors and spans, and in our central technical result, show that groubits correspond precisely to biunitary connections in $\mathbf{GpdAct_s}$[3]. We also give a 2-dimensional graphical programming language for groubits, and give a thorough development of its syntax and semantics. In Section 3 we give programs for state transfer, entanglement creation, teleportation and dense coding on networks of groubits, and verify these protocols using the rules of our abstract 2-dimensional syntax. We comment on the potential applicability of these protocols for message transfer and key distribution within networks of groubits. Further technical details on $\mathbf{GpdAct_s}$ and its quantization functor are given in an extended version of this paper [28].

1.3 Related work

Timeless networks. The concept of timeless networking and the possibility of timeout-free atomic message routing is due to Borrill and Karp [9], who also described the quantum properties of the technology. Our treatment here is inspired in part by their ideas, but does not follow the technical details of their approach.

Spekkens' toy model. A toy model for quantum phenomena has been developed by Spekkens and others [2, 11, 13, 26, 31] based on the *knowledge balance* principle, in which quantum-like effects arise by restricting an observer's ability to gain information about the state of a classical system. This principle can be seen as playing a role here, since groubits exhibit precisely such a balance between observable and unobservable states. Work on the toy model includes classical versions of several quantum procedures, including teleportation and dense coding which we also analyze here; furthermore, the low-level combinatorics are strongly similar in places (compare for example [31, Section I] with Figure 12 here.)

Our work goes beyond these results in a number of ways, including: identification of biunitary structures in $\mathbf{GpdAct_s}$ as a mathematical foundation; classification of these structures in terms of groubits; applications to timeless networks, key distribution and state

[3] See Section 1.3 for background on biunitaries.

transfer; the 2-dimensional high-level language for designing and verifying groubit programs; and the identification of a functorial mapping from our calculus to quantum theory. Also, we have a fundamentally different perspective: while the work cited above studies the toy model as a 'foil theory'—an exercise in quantum foundations which sheds light on the distinction between quantum and classical reality—our perspective is technological, focussed on writing and verifying programs for these hypothetical devices, which may be implementable and practically useful in the real world.

Groupoidification. Our work is close in spirit to the groupoidification programme developed by Baez, Morton and others [3, 4, 7, 23] from the combinatorial species of Joyal [20]; as here, they develop a 2-categorical groupoid-based model for quantum-like phenomena, equipped with a functorial mapping into traditional quantum theory. Yet there is a surprising disconnect: while their work is based on groupoids, spans, and spans of spans, ours is based on groupoids, free profunctors and spans. This technical distinction seems mild, yet is essential for our results, and we are not aware of a direct relationship between the settings.

Classical key distribution. Maurer [21] has suggested classical procedures for secure key distribution based on noisy communication channels. In his words, he drops the "apparently innocent assumption that, except for the secret key, the enemy has access to precisely the same information as the legitimate receiver". This is fundamentally different to our model, in which—just as in quantum key distribution—the "enemy" has access to the entire apparatus.

Biunitaries. Our main proof technique is the technology of *biunitaries* (see Section 2.3.) Introduced by Ocneanu [25] in 1989 and since developed by Jones, Morrison and others [18, 19, 22], they are a central tool in the classification of subfactors, a major research effort in pure mathematics. Biunitaries belong to the theory of *planar algebras*, which studies the linear representation theory of algebraic structures in the plane. The 2-dimensional syntax we use in this paper derives heavily from the work of this community. These planar algebra techniques have been used by the present authors and others [17, 27, 29, 33] to give a high-level language for quantum computation.

Unpublished work. Related ideas have been described by Bar and the second author in an unpublished note [5].

2 Foundations

2.1 Groudits and dits

Groudits. We begin with the definition of a groudit.

▶ **Definition 1.** A *groudit* $\mathcal{G}$ is a skeletal groupoid of the form $\mathbf{G} = \coprod_i G_i$, where G_i are finite groups, equipped for each $i \in \mathrm{Ob}(\mathbf{G})$ with bijections $\sigma_i, \tau_i : G_i \to \mathrm{Ob}(\mathbf{G})$.

Thinking about the consequences of this definition, we see that the underlying groupoid of a groudit is a disjoint union of n finite groups for some $n \in \mathbb{N}$, each with n elements. Note that the bijection data is not required to satisfy any properties, so groudits are easy to construct.

Just as classical bits are special cases of dits, so groubits are special cases of groudits.

Figure 1 Notation for states of a groubit and a bit.

▶ **Definition 2.** A *groubit* $\mathcal{B}$ is the groudit with identity bijections, and with underlying groupoid $\mathbf{B}$ defined as follows, where s, t are the source and target functions:

$$\mathrm{Ob}(\mathbf{B}) := \mathbb{Z}_2 \qquad \mathrm{Mor}(\mathbf{B}) := \mathbb{Z}_2 \times \mathbb{Z}_2 \qquad s, t := \mathbb{Z}_2 \times \mathbb{Z}_2 \xrightarrow{\pi_1} \mathbb{Z}_2 \tag{1}$$

Composition is defined as follows: $(a, b) \circ (a, c) := (a, b \oplus c)$.

So for $a, b \in \mathbb{Z}_2$, we write (a, b) to denote a morphism of type $a \to a$. Using the terminology of Section 1, we interpret a as the logical bit, and b as the internal bit. It follows from the composition law that the identity morphisms are of the form $(a, 0)$. For the bijection data, we exploit the fact that the groupoid is in a natural way the disjoint union of two copies of $\mathbb{Z}_2$, and so the bijections have the type $\sigma_i, \tau_i : G_i = \mathbb{Z}_2 \to \mathrm{Ob}(B) = \mathbb{Z}_2$. We choose all 4 of these bijections to be the identity.

For every protocol we give in this paper, we describe an implementation for an arbitrary groudit, and prove correctness at this general level. However, for informal discussions of groudit phenomena, and for the explicit traces of each protocol that we give throughout Section 3, we talk in terms of groubits.

Dits. We can also describe classical dits using groupoids.

▶ **Definition 3.** A *dit* $\mathcal{D}$ is a discrete skeletal groupoid $\mathbf{D}$ with d morphisms.

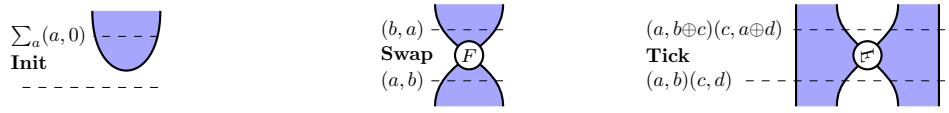
We recall that a groupoid is discrete when every morphism is an identity. For a dit $\mathcal{D}$, we write $[i]$ to denote a morphism $i \in \mathrm{Mor}(\mathbf{D})$. An ordinary classical bit is a dit with $d = 2$.

States. A *state* of a groudit or dit is a morphism in the corresponding groupoid. Our dynamics are nondeterministic, so after a protocol, the final state of a system is in general a multiset drawn from the set of states. We indicate these multisets with a sum notation, with coefficients drawn from $\mathbb{N}$.

In our graphical calculus, a groudit is represented by a blue region, and a classical dit by a yellow region. To indicate the state of the system at a given time, we draw a horizontal dashed line, and write the state to the left; see Figure 1.

Operations. In our graphical calculus we define *atomic operations*, and also *derived operations* which are built from the atomic operations. We summarize these here, and show explicitly how they act on groubits and bits. This notation is all that is required to follow the protocol traces illustrated in Section 3. In all our diagrams, time flows from bottom to top. All operations listed here map every input state to a nonempty multiset, meaning that they will not fail. That makes them suitable building blocks for a groudit programming language.

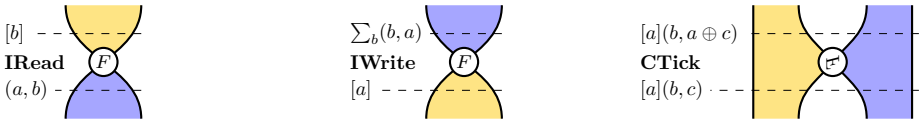
Atomic operations. In Figure 2 we list the atomic operations involving a bit and a groubit. Figure 2(a)–(c) shows the three groubit-only operations: **Swap** and **Tick** are deterministic, while **Init** creates a groubit in a nondeterministic logical state. Figure 2(c) uses a rotated



(a) Init **(b) Swap** **(c) Tick**

(d) Rand **(e) Erase** **(f) Read** **(g) Write**

Figure 2 Atomic groubit and bit operations.



(a) IRead **(b) IWrite** **(c) CTick**

(d) CTick **(e) Split**

Figure 3 Derived groubit and bit operations.

letter to label the vertex, since it is represented algebraically by a rotated version of Figure 2(b) under the dagger pivotal structure (see discussion below.)

Note that the result of performing two successive **Tick** operations between neighboring parties Alice and Bob, and Bob and Charlie, does not depend on the order of the operations; there is no race condition. Using the expression in Figure 2(c) this becomes a simple isotopy, a crucial feature of our 2-dimensional graphical calculus.

Figure 2(d)–(e) represents nondeterministic generation and erasure of a classical bit. Figure 2(f)–(g) give the basic interactions between a groubit and a bit: **Read** depicts the read-out of the logical state of a groubit, and **Write** depicts the initialization of a groubit with given logical bit and random internal bit.

Derived operations. In Figure 3 we list the derived operations **IRead**, **IWrite**, **CTick** and **Split**. Note that **CTick** comes in both left and right versions, distinguished by their images. We will see how they are defined in terms of atomic operations later in this section.

2.2 Graphical calculus

Definition. Our graphical calculus represents groupoids, free actions and spans. We begin with an informal definition of the 2-category formed by these structures. Throughout, we write '2-category' to refer to the weak structure, which is sometimes called 'bicategory'.

▶ **Definition 4.** The 2-category $\mathbf{GpdAct_s}$ is built from the following structures:

- *objects* are finite skeletal groupoids $\mathbf{G}$, $\mathbf{H}$, ...;

(a) A 2-morphism **(b)** A deformation

Figure 4 Examples of the graphical calculus.

- a *morphism* $S : \mathbf{G} \nrightarrow \mathbf{H}$ comprises, for any $a \in \mathrm{Ob}(\mathbf{G})$ and $b \in \mathrm{Ob}(\mathbf{H})$, a finite set $S_{a,b}$ equipped with commuting free left- and right-actions of $\mathrm{Aut}_{\mathbf{G}}(a)$ and $\mathrm{Aut}_{\mathbf{H}}(b)$ respectively;
- for morphisms $S, T : \mathbf{G} \nrightarrow \mathbf{H}$, a *2-morphism* $\sigma : S \Rrightarrow T$ is an *equivariant span*, comprising for all $a \in \mathrm{Ob}(\mathbf{G})$ and $b \in \mathrm{Ob}(\mathbf{H})$ a function $\sigma_{a,b} : S_{a,b} \times T_{a,b} \to \mathbb{N}$, such that for all $g \in \mathrm{Aut}_{\mathbf{G}}(a)$, $h \in \mathrm{Aut}_{\mathbf{H}}(b)$, $s \in S_{a,b}$ and $t \in T_{a,b}$ we have $\sigma_{a,b}(s,t) = \sigma_{a,b}(g.s.h, g.t.h)$.

Here $g.s.h \in S_{a,b}$ denotes the action on s by g on the left and h on the right; since these actions commute, this is well-defined. Note the requirement that these left- and right-actions are free, which is important to guarantee that our constructions are well-defined. In the main part of this paper we will work with these structures informally; we give a formal 2-categorical analysis in an extended version of this paper [28].

▶ **Definition 5.** For an equivariant span $\sigma : S \Rrightarrow T$, we define its *dagger* $\sigma^\dagger : T \Rrightarrow S$ as the converse: $\sigma^\dagger_{a,b}(t,s) := \sigma_{a,b}(s,t)$.

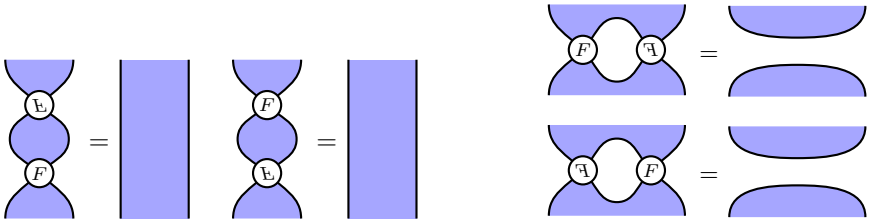
Graphical calculus. We use a 2-dimensional graphical calculus (see Figure 4(a)) to denote a 2-morphism σ in $\mathbf{GpdAct_s}$. This is the standard graphical calculus for 2-categories [30]: objects $\mathbf{G}$, $\mathbf{H}$ label the regions, morphisms $S, T : \mathbf{G} \nrightarrow \mathbf{H}$ label the wires, and 2-morphisms $\sigma : S \Rrightarrow T$ label the vertices. We often drop the labels; also, white regions will always correspond to the trivial groupoid $\mathbf{1}$ with one morphism.

Stacking these pictures vertically performs composition of spans, stacking them horizontally performs bimodule composition, and reflecting them about a horizontal axis corresponds to the dagger operation of Definition 5. In fact, $\mathbf{GpdAct_s}$ is a *dagger pivotal 2-category* [10, Section 2.1], giving immense freedom in the graphical calculus: one may reflect, rotate and deform parts of the pictures arbitrarily (holding the boundaries fixed), preserving equality. For example, since the images of Figure 4(a) and (b) are deformations of each other with constant boundary, they represent equal 2-morphisms.

2.2.0.1 Boundaries.

For every shaded region labelled by a skeletal groupoid $\mathbf{G}$, we have canonical boundaries drawn as follows:

$L^{\mathbf{G}} : \mathbf{1} \nrightarrow \mathbf{G}$ $R^{\mathbf{G}} : \mathbf{G} \nrightarrow \mathbf{1}$



(a) Vertical unitarity **(b)** Horizontal unitarity

Figure 5 The biunitarity equations.

We define these as the following free profunctors, for all objects $a \in \mathbf{G}$:

$$L^{\mathbf{G}}_{\bullet,a} := \mathrm{Aut}_{\mathbf{G}}(a) \qquad (\bullet, g, g') \mapsto gg' \tag{2}$$

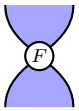
$$R^{\mathbf{G}}_{a,\bullet} := \mathrm{Aut}_{\mathbf{G}}(a) \qquad (g', g, \bullet) \mapsto g'g \tag{3}$$

That is, these boundaries are defined as the groupoid acting on itself, by left or right action. Using the pivotal structure, these boundaries give rise to the operations **Init**, **Erase** and **Split** as presented in Section 2.1.

2.3 Biunitaries

Biunitaries are important structures from the theory of planar algebras (see Section 1.3) which play an essential role in our calculus.

▶ **Definition 6.** In $\mathbf{GpdAct_s}$, a biunitary on a skeletal groupoid $\mathbf{G}$ is a unitary 2-morphism



(4)

fulfilling the equations depicted in Figure 5.

The source and target of a biunitary is the set $\mathrm{Mor}(\mathbf{G})$ of morphisms of the skeletal groupoid. So concretely, a biunitary is an automorphism of $\mathrm{Mor}(\mathbf{G})$ satisfying an algebraic condition. The following theorem determines this condition precisely.

▶ **Theorem 7.** *A biunitary on a skeletal groupoid* $\mathbf{G}$ *is a bijection* $F : \mathrm{Mor}(\mathbf{G}) \to \mathrm{Mor}(\mathbf{G})$ *such that for all* $a, b \in \mathrm{Ob}(\mathbf{G})$*, we have:*

$$|F(\mathrm{Aut}_{\mathbf{G}}(a)) \cap \mathrm{Aut}_{\mathbf{G}}(b)| = 1 \tag{5}$$

Proof. The equations of Figure 5(a) say that F is unitary, which means precisely that it acts as a permutation on $\mathrm{Mor}(\mathbf{G})$. The equations of Figure 5(b) are equivalent to the composite of Figure 6 being the identity.

This holds just when, for all $a, b \in \mathrm{Ob}(\mathbf{G})$ and for all $g \in \mathrm{Aut}_{\mathbf{G}}(a)$ and $g' \in \mathrm{Aut}_{\mathbf{G}}(b)$, there are unique $h \in \mathrm{Aut}_{\mathbf{G}}(a)$, $r \in \mathrm{Aut}_{\mathbf{G}}(b)$ with $s(F(h)) = b$ and $s(F^{-1}(r)) = a$ satisfying the following conditions:

$$gh^{-1}F^{-1}(r) = g$$

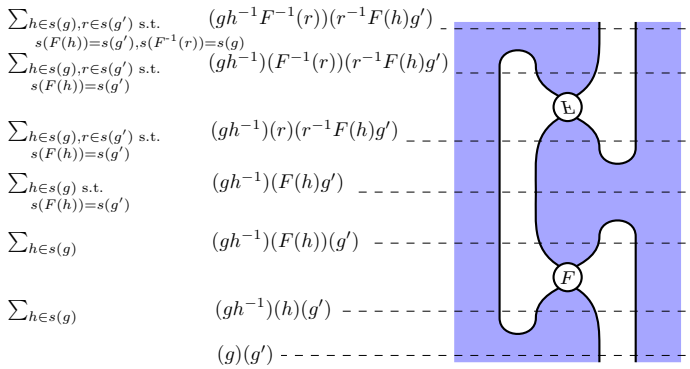


Figure 6 Verifying the action of a biunitary.

$$r^{-1}F(h)g' = g'$$

In other words for any two objects a, b in the groupoid there exists a unique pair $(h, r) \in \mathrm{Aut}_{\mathbf{G}}(a) \times \mathrm{Aut}_{\mathbf{G}}(b)$ such that $F(h) = r$. More concisely, $|F(\mathrm{Aut}_{\mathbf{G}}(a)) \cap \mathrm{Aut}_{\mathbf{G}}(b)| = 1$. ◀

Classification. We now classify biunitaries in terms of groudits. This shows that biunitaries are tractable algebraic objects.

▶ **Theorem 8.** *For a skeletal groupoid* $\mathbf{G}$*, groudits on* $\mathbf{G}$ *are in bijective correspondence with biunitaries on* $\mathbf{G}$*.*

Proof. Define a *balancer* ϵ for $\mathbf{G}$ to be a choice for all objects $a \in \mathrm{Ob}(\mathbf{G})$ of a bijection $\epsilon_a : \mathrm{Aut}_{\mathbf{G}}(a) \to \mathrm{Ob}(\mathbf{G})$. Clearly for any $b \in \mathrm{Ob}(\mathbf{G})$ we have

$$s(\epsilon_a^{-1}(b)) = a. \tag{6}$$

It is easy to see from the definition that a groudit is precisely a skeletal groupoid equipped with a pair of balancers. Given a balancer ϵ, we define functions ϵ_1, ϵ_2 as follows:

$$\epsilon_1 : \mathrm{Mor}(\mathbf{G}) \to \mathrm{Ob}(\mathbf{G}) \times \mathrm{Ob}(\mathbf{G}) \qquad \epsilon_1(g) := (sg, \epsilon_{sg}(g)) \tag{7}$$

$$\epsilon_2 : \mathrm{Ob}(\mathbf{G}) \times \mathrm{Ob}(\mathbf{G}) \to \mathrm{Mor}(\mathbf{G}) \qquad \epsilon_2(a, b) := \epsilon_a^{-1}(b) \tag{8}$$

We can show that ϵ_1 and ϵ_2 are inverse:

$$\epsilon_1(\epsilon_2(a, b)) = (s(\epsilon_a^{-1}(b)), \epsilon_{s(\epsilon_a^{-1}(b))}(\epsilon_a^{-1}(b)) \overset{(6)}{=} (a, \epsilon_a(\epsilon_a^{-1}(b))) = (a, b)$$

$$\epsilon_2(\epsilon_1(g)) = \epsilon_{s(g)}^{-1}(\epsilon_{s(g)}(g)) = g$$

We now give the first direction of the main bijective correspondence. Suppose ϵ, τ are balancers for $\mathbf{G}$. Then we define a biunitary $F_{\epsilon,\tau} : \mathrm{Mor}(\mathbf{G}) \to \mathrm{Mor}(\mathbf{G})$ as the following composite, where γ is the swap map for the cartesian product:

$$\mathrm{Mor}(\mathbf{G}) \xrightarrow{\epsilon_1} \mathrm{Ob}(\mathbf{G}) \times \mathrm{Ob}(\mathbf{G}) \xrightarrow{\gamma} \mathrm{Ob}(\mathbf{G}) \times \mathrm{Ob}(\mathbf{G}) \xrightarrow{\tau_2} \mathrm{Mor}(\mathbf{G})$$

Then a simple calculation shows the following:

$$F_{\epsilon,\tau}(g) = \tau_{\epsilon_{s(g)}(g)}^{-1}(s(g)) \in \mathrm{Aut}_{\mathbf{G}}(\epsilon_{s(g)}(g)) \tag{9}$$

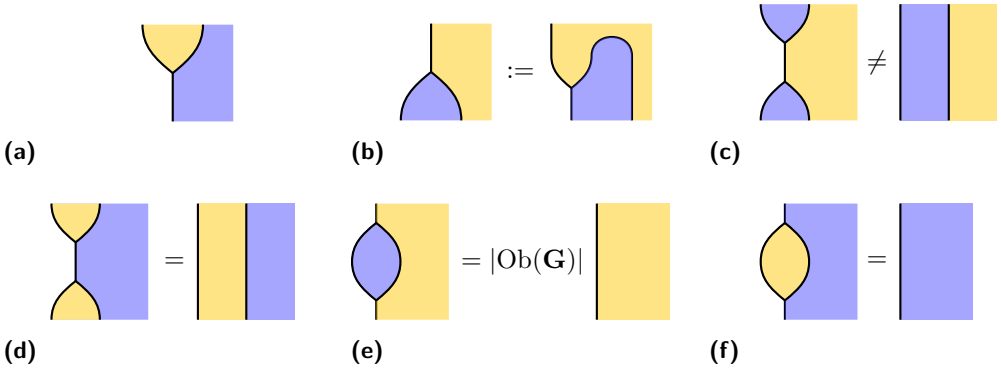


Figure 7 Building blocks for the measurement calculus.

By construction, $F_{\epsilon,\tau}$ is unitary, since it is a composite of bijections. To show it is biunitary, suppose now that $g, g' \in \mathrm{Aut}_{\mathbf{G}}(a)$ such that $s(F_{\epsilon,\tau}(g)) = s(F_{\epsilon,\tau}(g'))$. Then by equation (6), we have $\epsilon_{s(g)}(g) = \epsilon_{s(g')}(g')$, and since $s(g) = s(g') = a$ we therefore have $\epsilon_a(g) = \epsilon_a(g')$, and since ϵ_a is a bijection we have $g = g'$.

We now give the reverse direction of the main bijective correspondence. Given a biunitary $F : \mathrm{Mor}(\mathbf{G}) \to \mathrm{Mor}(\mathbf{G})$, we define balancers ϵ^F, τ^F for all $a \in \mathrm{Ob}(\mathbf{G})$ and $g \in \mathrm{Mor}(\mathbf{G})$ as

$$\epsilon^F_a(g) := s(F(g)) \qquad \tau^F_a(g) := s(F^{-1}(g)) \tag{10}$$

We must show that for all $a \in \mathrm{Ob}(\mathbf{G})$, $\epsilon^F_a, \tau^F_a : \mathrm{Aut}_{\mathbf{G}}(a) \to \mathrm{Ob}(\mathbf{G})$ are bijections. First, surjectivity. For any $b \in \mathrm{Ob}(\mathbf{G})$, using the biunitarity property (5), pick the unique morphism $g \in F(\mathrm{Aut}_{\mathbf{G}}(a)) \cap \mathrm{Aut}_{\mathbf{G}}(b)$. Then $F^{-1}(g) \in \mathrm{Aut}_{\mathbf{G}}(a)$ and $b = s(g) = s(F(F^{-1}(g))) = \epsilon^F_a(F^{-1}g)$. A similar proof shows surjectivity of τ^F_G. Next, injectivity. Suppose that $g, h \in \mathrm{Aut}_{\mathbf{G}}(a)$ with $\epsilon^F_a(g) = \epsilon^F_a(h)$; then $s(F(g)) = s(F(h))$. Then $F(g), F(h) \in F(\mathrm{Aut}_{\mathbf{G}}(a)) \cap \mathrm{Aut}_{\mathbf{G}}(s(F(g)))$. Then by the biunitarity property (5), we conclude that $F(g) = F(h)$ and therefore that $g = h$.

Finally, we show that the main correspondence is indeed bijective. In one direction, for a pair of balancers (ϵ, τ) with associated biunitary $F_{\epsilon,\tau}$ and $g \in \mathrm{Aut}_{\mathbf{G}}(a)$, then by (6) we have $\epsilon^{F_{\epsilon,\tau}}_a(g) = s(F_{\epsilon,\tau}(g)) = \epsilon_a(g)$ and similarly $\tau^{F_{\epsilon,\tau}}_a(g) = s(F^{-1}_{\epsilon,\tau}(g)) = \tau^F_a(g)$. In the other direction, given a biunitary F and $g \in \mathrm{Aut}_{\mathbf{G}}(a)$, we observe that $F_{\epsilon^F,\tau^F}(g) = (\tau^F_{\epsilon^F_a(g)})^{-1}(a)$. To show that this equals $F(g)$, we have to show that $\tau^F_{\epsilon^F_a(g)}(F(g)) = a$. And indeed, we have $\tau^F_{\epsilon^F_G(g)}(F(g)) = s(F^{-1}(F(g))) = s(g) = a$. ◀

2.4 Measurements

Syntax. In Section 2.1 we describe classical dits using discrete groupoids. In the graphical calculus we draw them as yellow regions, to distinguish them from groudits which we draw in blue. There is an important difference: while blue are equipped with a biunitary of the form (4), yellow regions are not equipped with any such structure.

Every groudit has its associated dit, with the logical states of the groudit corresponding to the elements of the dit. These interact via the 2-morphisms given in Figure 7(a) and (b). These are not physical elements of the groudit programming language (explaining why they do not appear in Section 2.1), but auxiliary mathematical structures that we will use to verify our groudit programs. In Figure 7(a) we begin with a groudit, and we read it to extract some classical data indicated by the yellow region; the groudit itself still exists.

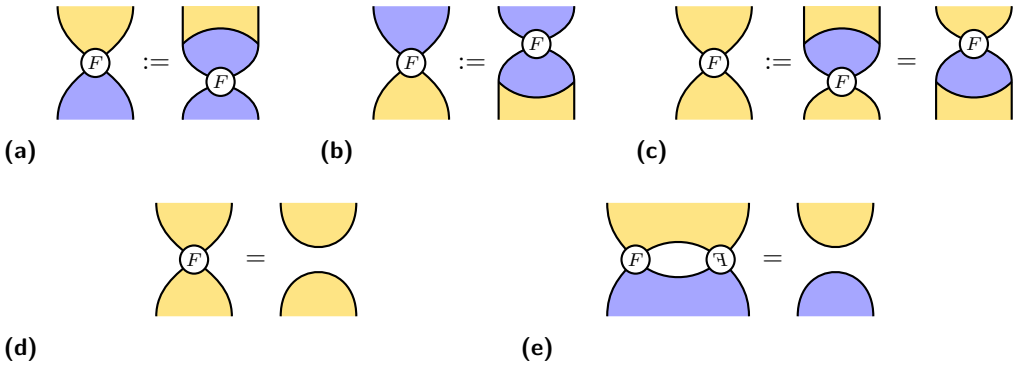


Figure 8 Yellow-blue and yellow-yellow versions of the biunitary.

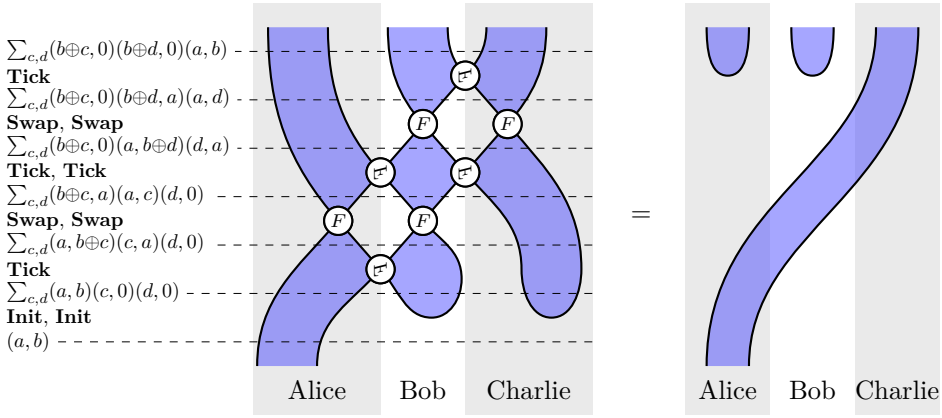
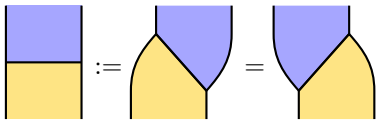


Figure 9 State transfer.

These building blocks are required to satisfy the axioms Figure 7(d), (e) and (f). By way of warning, Figure 7(c) shows a *nonequation* that is not satisfied in general.

Given the topological behaviour encoded in Figure 7(b), we can be relaxed about how we draw the interface between yellow and blue regions:



(11)

This gives us our composite **Write** operation; **Read** is the dagger of this. We also use this to define yellow-blue and yellow-yellow versions of the biunitary in Figure 8(a)–(c), which we require to satisfy equations Figure 8(d) and (e). These structures yield our composite operations **IRead**, **IWrite**, **LRead**, **RRead** and **CTick**.

Semantics. We suppose that the blue region represents a groudit $\mathcal{G}$ with underlying groupoid $\mathbf{G}$, and the yellow region represents a classical dit $\mathcal{B}$ with underlying groupoid $\mathbf{B}$, such that $\mathbf{B}$ is a discrete groupoid with the same set of objects as $\mathbf{G}$. We define the yellow-blue morphism $S : \mathbf{B} \nrightarrow \mathbf{G}$ as follows, where $\emptyset$ is the empty set:

$$S_{b,g} := \begin{cases} \mathrm{Aut}_{\mathbf{G}}(g) & \text{if } b = g \\ \emptyset & \text{otherwise} \end{cases} \tag{12}$$

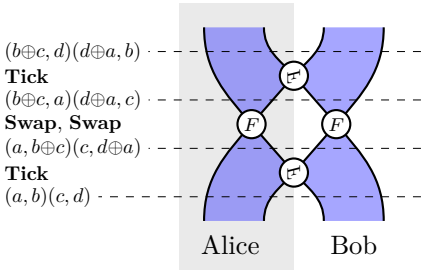


Figure 10 The basic state transfer repeating block.

The blue-yellow morphism $S^* : \mathbf{G} \nrightarrow \mathbf{B}$ is defined similarly. We define Figure 7(a) as follows, for all $a \in \mathrm{Ob}(\mathbf{G}) = \mathrm{Ob}(\mathbf{B})$ and $g \in \mathrm{Aut}_{\mathbf{G}}(a)$:

$$\text{Figure 7(a)} \qquad g \mapsto (a, g) \tag{13}$$

The span (13) is unitary, which explains equations Figure 7(d) and (f). These definitions satisfy the required equations.

▶ **Proposition 9.** *Using definitions* (12) *and* (13)*, the equations Figure 7(d)–(f) and Figure 8(d) and (e) all hold.*

3 Protocols

3.1 State transfer (Figures 9 and 10)

Overview. The state transfer protocol communicates a groubit down a linear chain of nodes, such that each node is connected to its neighbour with a link. Our mathematical treatment is closely related to a state transfer protocol for cluster-based quantum computers proposed previously by the authors [29]. The adjective *timeless* arises from a specific property of this protocol, which we examine below.

Program. The state transfer program is illustrated in Figure 9(b) for three parties, Alice, Bob and Charlie, arranged in a linear chain. Each party has a node, and separate links connect Alice and Bob, and also Bob and Alice, enabling **Tick** operations between connected parties. Alice has a groubit, which she would like to transfer to Charlie coherently; that is, preserving the internal state. The protocol is formed from repetitions of the *basic scheme* (see Figure 10), involving a **Tick** operation, two **Swap** operations, and a final **Tick**. In Figure 9(a) we use 2 copies of this basic building block, one between Alice and Bob, and one between Bob and Charlie. The generalization to arbitrary linear chains is clear.

Verification. The protocol is verified in the general case by observing that Figure 9(a) can be transformed into Figure 9(b) by applying the equations of Figure 5. On the left-hand side of Figure 9(a) we give an explicit program trace for the case of a $\mathbb{Z}_2 \sqcup \mathbb{Z}_2$ groubit, based on the lookup table in Section 2.1. The final state is $\sum_{c,d\in\mathbb{Z}_2}(b\oplus c, 0)(b\oplus d, 0)(a, b)$; by a simple change of variables it is clear that this equals $\sum_{c,d\in\mathbb{Z}_2}(c, 0)(d, 0)(a, b)$ as required.

Discussion. This protocol has certain limitations. While multiple messages can be sent from left-to-right along such a linear chain of nodes, if one attempts to send a message from right-to-left at the same time using a reflected version of the protocol, then both messages

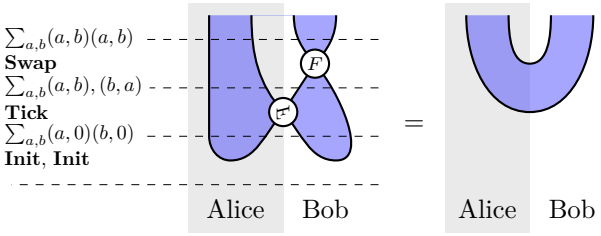


Figure 11 Entanglement creation.

will be corrupted. Of course, this could be overcome by having a pair of parallel chains, keeping left-to-right and right-to-left communications on separate tracks. Furthermore, we do not have a clear analysis of communication on a network with a more interesting topology.

Timelessness. A key property of this protocol is that it makes no use of timeouts, due to message atomicity properties that we now explore. This is desirable, since timeouts are a basic feature of the dominant TCP protocol for internet communication [16] which are the source of reliability issues in data centre environments [1, 9]. If any of the 4 operations of the scheme given in Figure 10 fail, Alice assumes that she retains ownership of the message, and is free to send it along another route of the network, or to return a failure message to the sender. Bob assumes ownership of the message if the final **Tick** occurs successfully.

3.2 Entanglement creation (Figure 11)

Overview. This is a procedure to create an 'entangled pair' of groubits. Entangled groubits are required for the dense coding and teleportation protocols described later.

Program. Alice and Bob each initialize a groubit. They then perform a **Tick** operation involving both their groubits. Finally, Bob performs a **Swap** operation.

Verification. Immediate by Figure 5(a).

Discussion. To implement this protocol, Alice and Bob must be connected by a link enabling the **Tick** operation.

3.3 Dense coding (Figure 12)

Overview. The dense coding procedure allows 2 classical bits to be transmitted between two parties, by transferring only 1 groubit. The parties must share an entangled pair of groubits, which could have been generated by the procedure discussed in Section 3.2.

Program. Alice begins with two classical bits, and Alice and Bob share an entangled pair of groubits. Alice begins by performing **CTick** operations (see Section 2.1) between her classical bits and her groubit, with a **Swap** operation in between. She then transfers the groubit to Bob, who performs a **Tick** operation between his two groubits, and then **IRead** operations on both groubits.

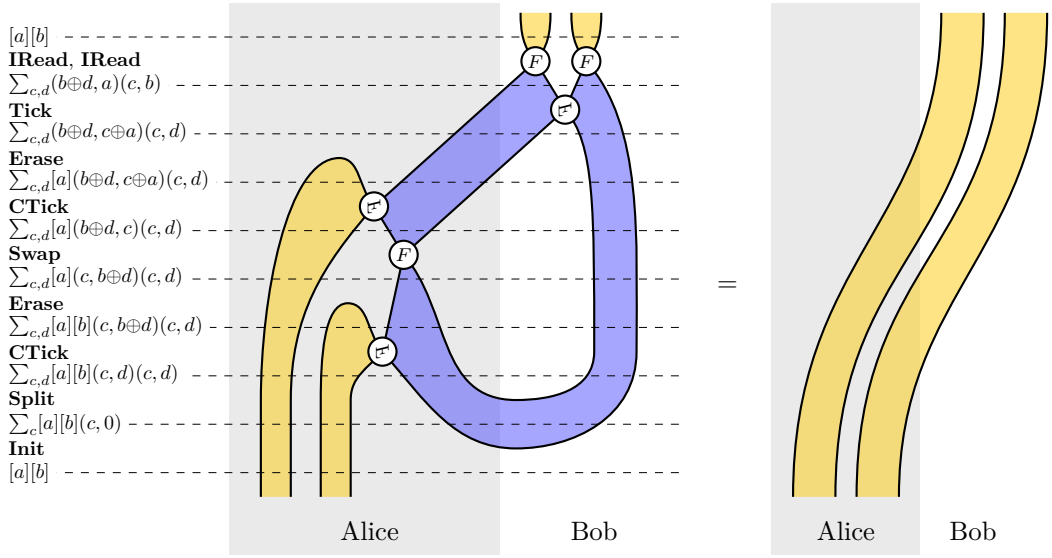


Figure 12 Dense coding.

Verification. To verify correctness of the program for general groudits, substitute the definitions of **IRead** and **CTick** in terms of the basic syntax, then apply equations from Figure 5 to cancel 3 pairs of adjacent F nodes.

Discussion. It may seem surprising that dense coding is possible, since although a groubit has 2 classical bits of memory, they cannot both be directly accessed; applying the **Read** operation (see Section 2.1) reveals the logical bit, but destroys the internal bit. The program requires passing a groubit from one agent to another; to implement this, agents could use the state transfer program described in Section 3.1.

Dense coding allows agents connected by a groubit network to double their effective data transfer rate, at the expense of consuming shared entanglement. It may be possible to use this for temporal load-balancing in a groubit data center. During times of low utilization, agents in the network perform entanglement creation (Section 3.2) to generate substantial numbers of shared entangled groubits. Later, when utilization of the data centre becomes high, these entangled groubits can be consumed to double the effective rate of data transfer.

3.4 Teleportation (Figure 13)

Overview. The teleportation procedure allows a groubit to be transported from one location to another, as long as those locations share an entangled groubit pair (see Section 3.2.)

Program. There are two parties, Alice and Bob. Alice starts with a groubit to be teleported, and Alice and Bob share between them an entangled pair of groubits. First, Alice performs a **Tick** operation on the groubit to be teleported. She then performs **Swap** operations on both of her groubits, then converts them into classical bits, which are transmitted to Bob by conventional means (for example, over the internet.) Bob then performs two **CTick** operations (see Figure 7), and performs **Erase** on the classical data received from Alice. The result is that Bob's groubit is now in the same state as Alice's was originally, both with respect to its logical and internal data.

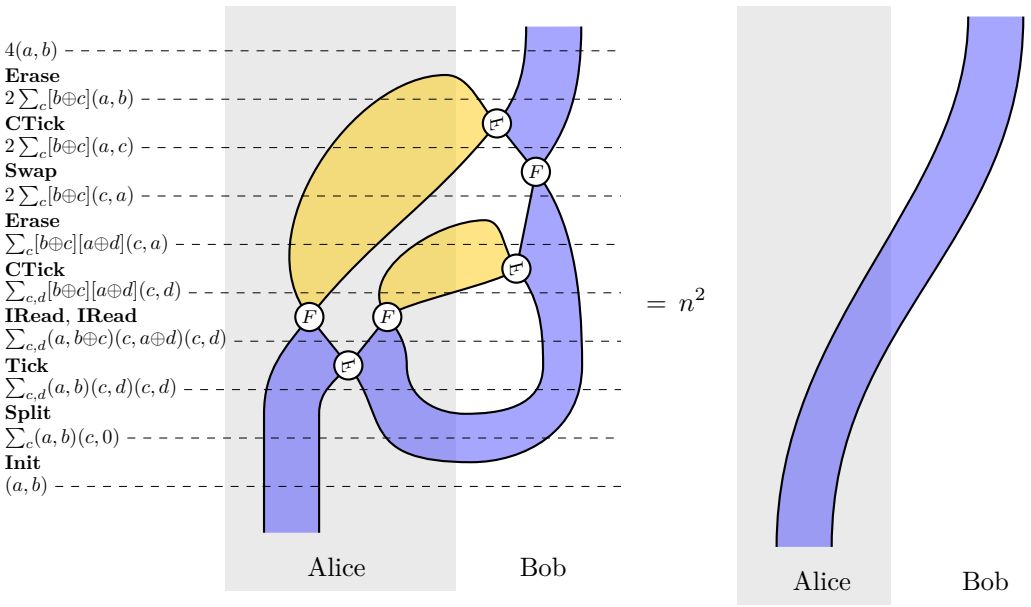


Figure 13 Teleportation.

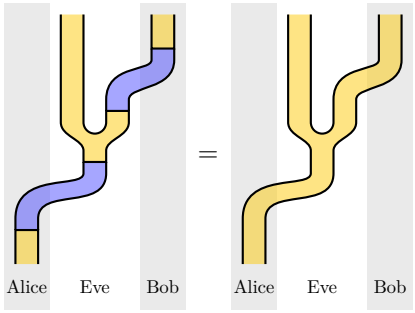
Verification. To verify the protocol in the general case, expand the **CTick** operations using the definitions from Figure 8(a) and (b), then apply equation Figure 8(e) twice. The result is the identity, up to two yellow bubbles, which count the different classical bits that Alice could have obtained.

Discussion. Teleportation may have an application for transferring groubits between separate groubit networks, which may only be connected via the internet. Of course, these data centres would have to be furnished with a sufficient supply of entangled groudits.

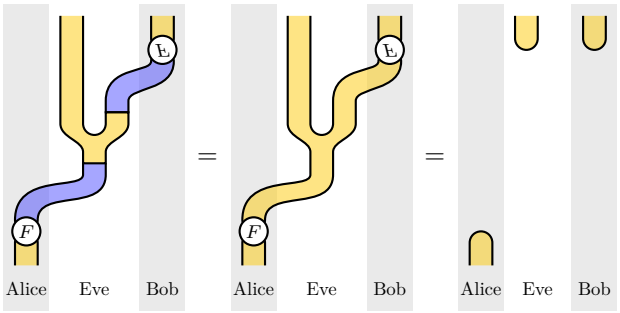
3.5 Key distribution (Figures 14 and 15)

Overview. Quantum key distribution (QKD) [15] is one of the most important protocols in quantum information. Here we describe a classical analogue which can operate on networks of groudits. The inability of the eavesdropper to read both the logical and internal state of a groubit is exploited to enable the effect. An analysis of QKD using a related graphical calculus has also been performed by Coecke and Perdrix [12]. We focus here on BB84-style QKD [6]; by dagger pivotality, the E91 variant [14] has a similar analysis (see Figure 15.)

Program. The basic setup of our key distribution protocol is given in Figure 15(a), and is similar to the BB84 QKD protocol [6]. Alice and Bob have an authenticated public classical channel, and a groubit channel, which are both accessible by an adversary Eve. Alice begins with a classical bit, and chooses at random to encode it into a groubit using $\alpha = \mathbf{Write}$ or $\alpha = \mathbf{IWrite}$. She sends the groubit to Bob, perhaps using a state transfer algorithm (see Section 3.1), but it is intercepted by Eve, who chooses to decode the message using either $\eta = \mathbf{Read}$ or $\eta = \mathbf{IRead}$; having received a classical bit she copies it, and re-encodes a groubit using $\eta^\dagger$, which she sends to Bob. When Bob receives the groubit, he decodes it using $\beta = \mathbf{Read}$ or $\beta = \mathbf{IRead}$.

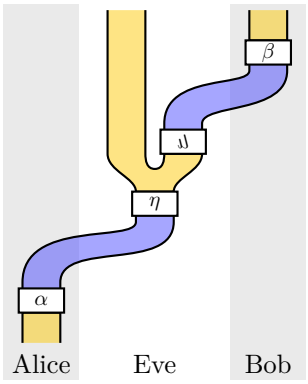


(a) All agents choose the same operation.

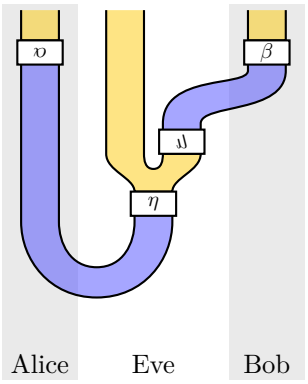


(b) Some agents choose a different operation.

Figure 14 Verification of BB84 quantum key distribution.



(a) BB84-like protocol.



(b) E91-like protocol.

Figure 15 Key distribution protocols.

Verification. The analysis proceeds in just the same way as for the traditional BB84 procedure. If Alice, Bob and Eve all choose the same operation ($\alpha = \eta^\dagger = \beta^\dagger$), then it is as if Alice's choice of initial bit is copied to Eve and Bob. We analyze this scenario in Figure 14(a), where we choose $\alpha = \eta^\dagger = \beta^\dagger =$ **Write**; using the equations of Figure 7, the equation can be verified. On the other hand, if any of the 3 parties do not choose the same operation, the diagram disconnects. We analyze this in Figure 14(b); using the equations of Figures fig:measurement and 8, and in particular Figure 8(d), this chain of equalities can also be shown, leading us to conclude that all parties receive uncorrelated random bits.

Discussion. This protocol may have real-world relevance, either for key distribution within an insecure data centre based on groubit networks, or on a larger scale. Our analysis here cannot be considered a full security proof; just as with genuine quantum key distribution, there are many compounding details that would affect the real security of the procedure.

Acknowledgements. We are grateful to Paul Borrill and Alan Karp for conversations about timeless networking, Steve Vickers for many helpful comments on an earlier version of this paper, and to Krzysztof Bar for substantial discussions about groupoid semantics and a classical model for key distribution.

References

1 Akintomide Adesanmi and Lotfi Mhamdi. Controlling TCP Incast congestion in data centre networks. In *ICCW 2015*. IEEE, 2015. doi:10.1109/iccw.2015.7247446.

2 Miriam Backens and Ali Nabi Duman. A complete graphical calculus for Spekkens' toy bit theory. *Foundations of Physics*, 46(1):70–103, 2015. doi:10.1007/s10701-015-9957-7.

3 John Baez and James Dolan. From finite sets to Feynman diagrams. In Björn Engquist and Wilfried Schmid, editors, *Mathematics Unlimited*, pages 29–50. Springer, 2001. arXiv:math/0004133.

4 John Baez, Alexander Hoffnung, and Christopher Walker. HDA VII: Groupoidification. *Theory and Applications of Categories*, 24(18):489–553, 2010. arXiv:0908.4305.

5 Krzysztof Bar and Jamie Vicary. Groupoid semantics for thermal computing, 2014. arXiv:1401.3280.

6 Charles H. Bennett and Gilles Brassard. Quantum public key distribution. *IBM Tech. Disc. Bul.*, 28:3153–3163, 1985.

7 Francois Bergeron, Gilbert Labelle, Pierre Leroux, and Margaret Readdy. *Combinatorial Species and Tree-like Structures*. Cambridge University Press (CUP), 1997. doi:10.1017/cbo9781107325913.

8 Daniel J. Bernstein. Introduction to post-quantum cryptography. In *Post-Quantum Cryptography*, pages 1–14. Springer Nature, 2009. doi:10.1007/978-3-540-88702-7_1.

9 Paul Borrill. The timeless datacentre. *Stanford Colloquium on Computer Systems*, 2016. YouTube:IPTlTmH-YvQ.

10 Nils Carqueville and Ingo Runkel. Orbifold completion of defect bicategories. *Quantum Topology*, 7(2):203–279, 2016. doi:10.4171/qt/76.

11 Bob Coecke, Bill Edwards, and Robert W. Spekkens. Phase groups and the origin of non-locality for qubits. *ENTCS*, 270(2):15–36, 2011. doi:10.1016/j.entcs.2011.01.021.

12 Bob Coecke and Simon Perdrix. Environment and classical channels in categorical quantum mechanics. *LMCS*, 8(4), 2012. doi:10.2168/lmcs-8(4:14)2012.

13 Leonardo Disilvestro and Damian Markham. Quantum protocols within Spekkens' toy model. *Physical Review A*, 95(5), 2017. doi:10.1103/physreva.95.052324.

14 Artur Ekert. Quantum cryptography based on Bell's theorem. *Physical Review Letters*, 67:661, 1991. doi:10.1103/PhysRevLett.67.661.

15 Romain Alléaume et al. Using quantum key distribution for cryptographic purposes: a survey. *TCS*, 560(1):62–81, 2014. arXiv:quant-ph/0701168.

16 Sami Iren, Paul D. Amer, and Phillip T. Conrad. The transport layer: tutorial and survey. *ACM Computing Surveys*, 31(4):360–404, 1999. doi:10.1145/344588.344609.

17 Arthur Jaffe, Zhengwei Liu, and Alex Wozniakowski. Holographic software for quantum networks, 2016. URL: https://arxiv.org/abs/1605.00127.

18 Vaughan F. R. Jones. Planar algebras, I, 1999. arXiv:math/9909027.

19 Vaughan F. R. Jones, Scott Morrison, and Noah Snyder. The classification of subfactors of index at most 5. *Bull. Amer. Math. Soc.*, 51(2):277–327, 2013. doi:10.1090/s0273-0979-2013-01442-3.

20 André Joyal. Une théorie combinatoire des séries formelles. *Advances in Mathematics*, 42(1):1–82, 1981. doi:10.1016/0001-8708(81)90052-9.

21 Ueli M. Maurer. Protocols for secret key agreement by public discussion based on common information. *IEEE Transactions on Information Theory*, 39(3):733–742, 1993. doi:10.1007/3-540-48071-4_32.

22 Scott Morrison and Emily Peters. The little desert? Some subfactors with index in the interval $(5, 3 + \sqrt{5})$. *International Journal of Mathematics*, 25(08):1450080, 2014. doi:10.1142/s0129167x14500803.

23 Jeffrey Morton. Categorified algebra and quantum mechanics. *Theory and Applications of Categories*, 16(29):785–854, 2006. arXiv:math/0601458.

24 Michael Nielsen and Isaac Chuang. *Quantum Computation and Quantum Information*. CUP, 2009. doi:10.1017/cbo9780511976667.

25 Adrian Ocneanu. Quantized groups, string algebras, and Galois theory for algebras. In *Operator Algebras and Applications*, pages 119–172. CUP, 1989. doi:10.1017/cbo9780511662287.008.

26 Matthew F. Pusey. Stabilizer notation for Spekkens' toy theory. *Foundations of Physics*, 42(5):688–708, 2012. doi:10.1007/s10701-012-9639-7.

27 David Reutter and Jamie Vicary. Biunitary constructions in quantum information, 2016. arXiv:1609.07775.

28 David Reutter and Jamie Vicary. A classical groupoid model for quantum networks, 2017. arXiv:1707.00966.

29 David Reutter and Jamie Vicary. Shaded tangles for the design and verification of quantum programs, 2017. arXiv:1701.03309.

30 Peter Selinger. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pages 289–355. Springer, 2010. doi:10.1007/978-3-642-12821-9_4.

31 Robert W. Spekkens. Evidence for the epistemic view of quantum states: A toy theory. *Physical Review A*, 75(3), 2007. doi:10.1103/physreva.75.032110.

32 Umesh Vazirani and Thomas Vidick. Robust device independent quantum key distribution. In *ITCS*, 2014.

33 Jamie Vicary. Higher semantics of quantum protocols. In *Proceedings of LICS*, 2012. doi:10.1109/lics.2012.70.

34 Yongwang Zhao, Zhibin Yang, and Dianfu Ma. A survey on formal specification and verification of separation kernels. *Frontiers of Computer Science*, 2017. doi:10.1007/s11704-016-4226-2.

A 2-Categorical Approach to Composing Quantum Structures*

David Reutter[1] and Jamie Vicary[2]

1 **Department of Computer Science, University of Oxford, UK**
david.reutter@cs.ox.ac.uk
2 **Department of Computer Science, University of Oxford, UK**
jamie.vicary@cs.ox.ac.uk

Abstract

We present an infinite number of construction schemes for quantum structures, including unitary error bases, Hadamard matrices, quantum Latin squares and controlled families, many of which have not previously been described. Our results rely on the type structure of biunitary connections, 2-categorical structures which play a central role in the theory of planar algebras. They have an attractive graphical calculus which allows simple correctness proofs for the constructions we present. We apply these techniques to construct a unitary error basis that cannot be built using any previously known method.

1998 ACM Subject Classification G.2.1 Combinatorics, F.1.1 Models of Computation

Keywords and phrases quantum constructions, 2-category, graphical calculus, planar algebra

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.20

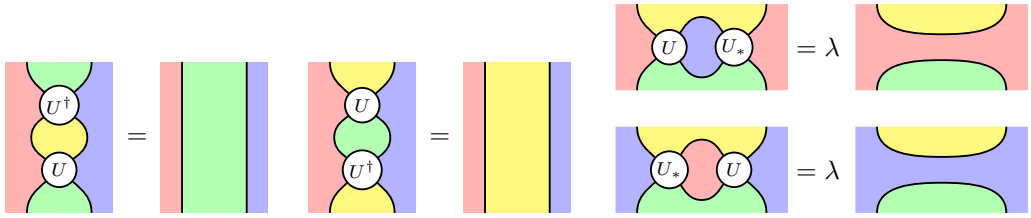
1 Introduction

Biunitary connections (or simply *biunitaries*) were introduced by Ocneanu [39] in 1989, and have since been developed by Jones, Morrison and others [23, 24, 34] as a central tool in the classification of subfactors. They belong to the theory of *planar algebras*, an area of mathematics related to 2-category theory which studies the linear representation theory of algebraic structures in the plane. We can describe a biunitary informally as a planar algebra element U with two inputs and two outputs, drawn below and above the vertex respectively, which is *vertically unitary* (Figure 1(a)), and which is *horizontally unitary* up to a scalar factor λ (Figure 1(b)).

For us, diagrams of this sort represent simple linear algebra data: regions are labelled by indexing sets, and wires and vertices are labelled by indexed families of finite-dimensional Hilbert spaces and linear maps, respectively.[1] Blank regions correspond to the trivial indexing set. In concrete terms, a biunitary therefore comprises a family of linear maps satisfying some algebraic properties.

The *type* of a biunitary is the shading pattern which surrounds the vertex. We show in Section 2 that a variety of structures in quantum information theory correspond exactly

* An extended version of this paper can be found at [44], https://arxiv.org/abs/1609.07775.
[1] Formally this is a common generalization of the tensor [23, Example 2.6] and spin model [23, Example 2.8] planar algebras, corresponding to a fragment of the monoidal 2-category **2Hilb** [4]. However, our exposition will be elementary, and we will not assume knowledge of these ideas.



(a) Vertical unitarity **(b)** Horizontal unitarity

Figure 1 The biunitarity equations.

(a) Hadamard matrix (Had) **(b)** Unitary error basis (UEB) **(c)** Quantum Latin square (QLS) **(d)** Controlled family of Hadamards (Had*)

Figure 2 Biunitary types for a variety of quantum structures.

to biunitaries of particular types. Some important examples are given in Figure 2.[2] In the rightmost image, we see that the notation is 3-dimensional, with the blue sheet lying beneath the yellow sheet. Rotations by a quarter-turn, and reflections about the horizontal or vertical axes, preserve the given interpretations in terms of quantum structures.

Some of these characterizations are already known: *Hadamard matrices*[3] were characterized by Jones as biunitaries with alternating shaded and unshaded regions [23], and *unitary error bases*[4] were characterized by the second author as biunitaries with one shaded and three unshaded regions [53,54]. Here we show that *quantum Latin squares*[5] can be characterized as biunitaries with two adjacent shaded regions and two adjacent unshaded regions. We also show that controlled families can be described by adding an additional shaded region in a certain way; in Figure 2(d), we illustrate one application of this idea to give a biunitary characterization of a controlled family of Hadamard matrices.

Our main results are based on the simple fact that the *diagonal* composite of two biunitaries is again biunitary. We show in Section 3 that, given the description of quantum combinatorial structures in terms of biunitaries as summarized above, one can immediately write down a large number of schemes for the construction of certain quantum structures from others. We give some examples in Figure 3; note that the biunitaries are connected diagonally in each case, as required.

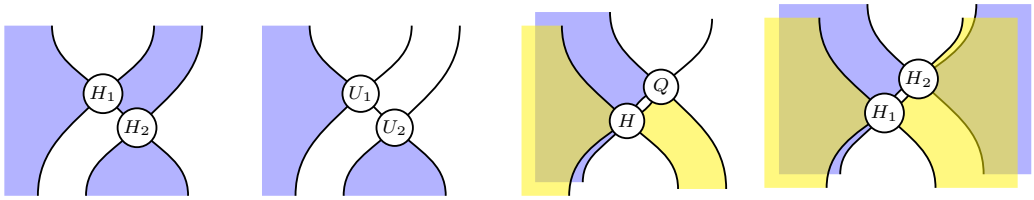
---

[2] Note that some of the inputs or outputs of the biunitary may in general be composite wires. For example, in Figure 2(c) the first input is composite, and Figure 2(d) the first input and second output are composite.

[3] A *Hadamard matrix* is a square complex matrix with entries of modulus 1, which is proportional to a unitary matrix. Fundamental structures in quantum information, they are central in the theories of mutually unbiased bases, quantum key distribution, and many other phenomena [18]. See Definition 6.

[4] A *unitary error basis* is a basis of unitary operators on a finite-dimensional Hilbert space, orthogonal with respect to the trace inner product. They provide the basic data for quantum teleportation, dense coding and error correction procedures [31, 49, 55]. See Definition 8.

[5] A *quantum Latin square* [36] is a square grid of vectors in a finite-dimensional Hilbert space, such that every row and every column is an orthonormal basis. They generalize Latin squares. See Definition 10.



(a) Had + Had $\rightsquigarrow$ QLS **(b)** UEB + UEB $\rightsquigarrow$ QLS **(c)** Had* + QLS $\rightsquigarrow$ UEB **(d)** Had* + Had* $\rightsquigarrow$ Had

Figure 3 Some biunitary composites of arity 2.

We explain each of these constructions briefly. Figure 3(a) gives a way to combine two Hadamard matrices to produce a quantum Latin square, generalizing a known construction.[6] (Note that the wires terminating near the upper-right of Figure 3(a) are interpreted as a *single* composite wire for the purpose of identifying it as having the basic quantum Latin square type of Figure 2(c), a method that we use repeatedly, and motivate formally with bracketings in Theorem 16.) Figure 3(b) gives a procedure for combining two unitary error bases to yield a quantum Latin square, a construction we believe to be new. Figure 3(c) combines a controlled family of Hadamard matrices and a quantum Latin square to give a unitary error basis, recovering the quantum shift-and-multiply construction [36, Definition 18]. In Figure 3(d), two families of Hadamard matrices are combined to produce a single Hadamard matrix, recovering a construction of Hosoya and Suzuki [21, Section 1] which generalizes a construction of Diţă [17, Section 4]. These constructions can of course be iterated; for example, combining the constructions of Figures 3(a) and 3(c) gives a way to combine a controlled family of Hadamard matrices and two further Hadamard matrices to produce a single unitary error basis, again a new construction.

In all these cases, correctness of the construction follows immediately from the type-theoretic structure (that is, the shading pattern) of the diagram, relying only on diagonality of the composition; no further details need to be checked. Our approach therefore offers advantages even for those constructions that are already known, since the traditional proofs of correctness are nontrivial. To emphasize this point we compare our graphical techniques to traditional methods, in which constructions are defined using tensor notation. For example, the construction of Figure 3(c) would traditionally be written as follows [36, Definition 18], where $U_{ab,c,d}$ is the (c,d)th matrix entry of the (a,b)th element of the unitary error basis, $Q_{b,d,c}$ is the coefficent of $|c\rangle$ in the (b,d)th position of the quantum Latin square, and $H^b_{a,d}$ is the (a,d)th coefficient of the bth Hadamard matrix:

$$U_{ab,c,d} := H^b_{a,d}\, Q_{b,d,c} \tag{1}$$

It is not trivial to write down correct expressions of this form, and to show that this indeed defines a unitary error basis requires a calculation of several lines [36, Theorem 20] that invokes the distinct algebraic properties of the tensors $Q_{b,d,c}$ and $H^b_{a,d}$. In contrast, in our new approach, it would be easy to discover this construction by considering all ways the basic components can be diagonally composed; correctness is immediate, and all algebraic properties are subsumed by the single concept of biunitarity. Nonetheless, expression (1) can be immediately read-off from the form of the biunitary composite.

[6] When both Hadamard matrices are the same, this agrees with a known construction of a quantum Latin square from a single Hadamard matrix [36, Definition 10].

(a) Had + Had + Had ⇝ UEB **(b)** Had** + QLS + QLS + UEB ⇝ UEB

Figure 4 Some biunitary composites of arities 3 and 4.

Higher-arity constructions can also be described, such as those given in Figure 4. Both of these we believe to be new. In Figure 4(a), arising as a consequence of the constructions of Figures 3(a) and 3(c), three Hadamard matrices combine to produce a unitary error basis, an elegant construction which we believe to be new.[7] In Figure 4(b), which does not arise as a consequence of lower-arity constructions, we combine a double-controlled family of Hadamard matrices (H), two quantum Latin squares (Q_1, Q_2) and a unitary error basis (U) to produce a new unitary error basis. While the first example is simple and elegant, the second example is indicative of the more complex constructions our technique can produce. Further complex examples are given in Figures 10.

For unitary error bases, we illustrate all constructions of arities 2 and 3 that arise from our methods, and we give examples of constructions of arities 4 and 8. Furthermore, in an extended version of this paper [44], we show that our methods give rise to an infinite family of logically independent constructions, none of which factor through any simpler construction between Hadamard matrices, unitary error bases, quantum Latin squares and controlled families thereof.

Finally, in the extended version we use the 4-fold composite of Figure 4(b) to produce a unitary error basis on an 8-dimensional Hilbert space, which we show cannot be produced by the two known UEB construction methods—algebraic, and quantum shift-and-multiply—even up to equivalence. This is a proof of principle that the biunitary methods we propose can give rise to genuinely new quantum structures.

Significance. Hadamard matrices and unitary error bases provide the mathematical foundation for an extremely rich variety of quantum computational phenomena, amongst them the study of mutually unbiased bases, quantum key distribution, quantum teleportation, dense coding and quantum error correction [18, 29, 31, 49, 55]. Nevertheless their general structure is notoriously difficult to understand; in dimension n, Hadamard matrices have only been classified up to $n = 5$ [50, 52], and the general structure of unitary error bases is virtually unknown for $n > 2$. Quantum Latin squares have been introduced much more recently [7,35,36], generalizing classical Latin squares which have a wide range of applications in classical and quantum information [10, 33, 48].

By unifying these quantum structures as special cases of the single notion of biunitary, and providing simple type-theoretical tools to understand the intricate interplay between them, we unify several already-known and seemingly-unrelated constructions [7,17,21,36,55],

[7] When all three Hadamard matrices are the same, this agrees with a known construction of a unitary error basis from a single Hadamard matrix [36, Definition 33] which we believe to be folklore.

uncover an infinite number of new constructions, and produce novel, concrete examples. These new tools may lead to further progress in questions of classification and applications of Hadamard matrices, unitary error bases and quantum Latin squares, and perhaps move us closer to full classification results for these important structures.

On the other hand, biunitaries are central tools in the study and classification of subfactors [23,24,34,39,43], a highly significant activity in the theory of von Neumann algebras. We hope that our work leads to the development of further connections between subfactor theory and quantum computation.

1.1 Related work

Quantum constructions. As well as producing a number of new constructions, our methods encompass and unify several constructions from the literature.

- The *Hadamard method* [36, Definition 33], believed to be folklore, which produces a unitary error basis from a single Hadamard matrix. In Figure 4(a) we give a new generalization, in which three Hadamard matrices produce a unitary error basis.
- The method given in [7, Definition 2.3] and [36, Definition 10], which produces a quantum Latin square from a single Hadamard matrix. In Figure 9(a) we give a new generalization, in which two Hadamard matrices produce a quantum Latin square.
- Werner's *shift-and-multiply construction* [55] which produces a unitary error basis from a family of Hadamard matrices and a Latin square. This is a special case of the quantum shift-and-multiply construction discussed below.
- The *quantum shift-and-multiply construction* due to Musto and the second author [36, Definition 18] which produces a unitary error basis from a family of Hadamard matrices and a quantum Latin square. We give a biunitary description in Figure 9(f).
- *Diţă's construction* [17, Section 4], which produces a Hadamard matrix from a Hadamard matrix and a family of Hadamard matrices, and is widely used [17,32,38,52]. We give a biunitary description in Figure 9(d).
- *Hosoya's and Suzuki's construction* [21, Section 1], which produces a Hadamard matrix from two families of Hadamard matrices. We give a biunitary description in Figure 9(c).

However, there are many known constructions which are beyond our methods. For unitary error bases, we do not know a biunitary characterization of Knill's algebraic construction [30], and for Hadamard matrices we cannot account for many of the varied construction methods [20, 32, 35, 42, 50, 51, 56] which make use of non-compositional structure that is out of reach of the biunitary approach.

Biunitary connections and planar algebras. Biunitary connections were introduced by Ocneanu in 1989 in terms of *paragroups* [39] in an attempt to better understand the combinatorial structures arising in subfactor theory, a branch of the theory of von Neumann algebras. In 1999, Jones introduced the theory of planar algebras [23] and with it the modern graphical formulation of biunitarity used in this paper.

The relation between Hadamard matrices and von Neumann algebras predates the notion of biunitarity and can be traced back to Popa's commuting squares [43] (later shown to be equivalent to biunitarity [26]) and the statistical-mechanical spin models of Jones [23, 25]; there is a significant literature on the interplay between Hadamard matrices, planar algebras and subfactors [20, 37, 38]. Quantum Latin squares appear under the name *magic bases* in a Hopf algebraic approach to subfactor theory by Banica and others [5–7]; however, their biunitary characterization does not seem to have been written down. Unitary error bases were characterized in terms of biunitaries by the second author [53, 54].

Categorical quantum mechanics. This work builds on the programme of categorical quantum mechanics, initiated by Abramsky and Coecke [1] and developed by them and others [2, 3, 11–15, 28, 46], which uses monoidal categories with duals to provide a high-level syntax for quantum information flow. It was shown by the second author that these ideas can be extended to a higher categorical setting [53, 54], developing the work of Baez on a categorified notion of Hilbert space [4], a perspective we use here. The key advantage of this approach is that the notion of Frobenius algebra, used in the monoidal category setting to describe classical information, is no longer needed. While our results could in principle be translated back into the language of Frobenius algebras, they would lose their simplicity and power. In this sense, the current work serves as an advertisement for the essential role that higher category theory can play in quantum information theory.

1.2 Notations and conventions

We denote the n-element set $\{1, \ldots, n\}$ by $[n]$. The letters $a, b, d, e, f, g, h, i, j, k, r, s$ are used to denote indices, the letters n, m, p, q are used to denote dimensions. We use the following shorthand notations to refer to sets of quantum structures:

- UEB_n is the set of n-dimensional unitary error bases;
- QLS_n is the set of n-dimensional quantum Latin squares;
- Had_n is the set of n-dimensional Hadamard matrices;
- For $\mathrm{X} \in \{\mathrm{UEB}_n, \mathrm{QLS}_n, \mathrm{Had}_n\}$, $\mathrm{X}^{p_1,\ldots,p_k}$ is the set of lists of quantum structures of type X controlled by indices in $[p_1]$, $[p_2]$, $\ldots$, $[p_k]$.

For example, $\mathrm{UEB}^{n,p}_{n^2m}$ is the set of lists of n^2m-dimensional unitary error bases, controlled by indices valued in $[n]$ and $[p]$.

2 Biunitarity

In Section 2.1 we introduce our formalism, and give the definition of biunitarity. In Section 2.2 we recall the biunitary characterizations of Hadamard matrices and unitary error bases, and give new biunitary characterizations of quantum Latin squares and controlled families.

2.1 Mathematical foundations

The graphical calculus for describing composition of multilinear maps was proposed by Penrose [41], and is today widely used in a range of areas [2, 11, 27, 40, 47]. In this scheme, wires represent Hilbert spaces and vertices represent linear maps between them, with wiring diagrams representing composite linear maps. For example, given linear maps $A : W \otimes H \otimes J \to L \otimes M \otimes R$ and $B : V \to H \otimes J$, we can describe a composite linear map $V \otimes W \to L \otimes M \otimes R$ graphically as shown in Figure 7(a).

In this article we use a generalized calculus that involves *regions*, as well as wires and vertices. This is an instance of the graphical calculus for monoidal 2-categories [8, 9, 22, 45] applied to the 2-category[8] of finite-dimensional 2-Hilbert spaces [4]. The 2-category of 2-Hilbert spaces can be described as follows [19, 53]:

- objects are natural numbers $n, m, \ldots$;
- 1-morphisms $n \to m$ are $m \times n$-matrices of finite-dimensional Hilbert spaces (Figure 5(a));
- 2-morphisms are matrices of linear maps (Figure 5(b)).

$$\begin{pmatrix} H_{11} & \cdots & H_{1n} \\ \vdots & \ddots & \vdots \\ H_{m1} & \cdots & H_{mn} \end{pmatrix}$$

(a) A 1-morphism $H : n \to m$.

$$\begin{pmatrix} H_{11} \xrightarrow{\phi_{11}} H'_{11} & \cdots & H_{1n} \xrightarrow{\phi_{1n}} H'_{1n} \\ \vdots & \ddots & \vdots \\ H_{m1} \xrightarrow{\phi_{m1}} H'_{m1} & \cdots & H_{mn} \xrightarrow{\phi_{mn}} H'_{mn} \end{pmatrix}$$

(b) A 2-morphism $\phi : H \Rightarrow H'$.

Figure 5 The 1- and 2-morphisms in the 2-category of 2-Hilbert spaces.

$$\phi : \begin{pmatrix} J_1 \\ J_2 \end{pmatrix} \Rightarrow \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \end{pmatrix} \circ \begin{pmatrix} K_1 \\ K_2 \\ K_3 \end{pmatrix}$$

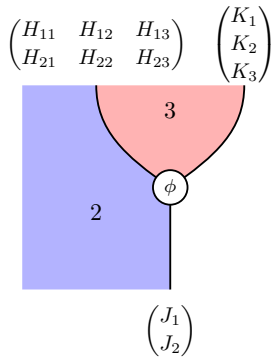
(a) A 2-morphism ϕ.

$$\begin{pmatrix} J_1 \xrightarrow{\phi_1} (H_{11} \otimes K_1) \oplus (H_{12} \otimes K_2) \oplus (H_{13} \otimes K_3) \\ J_2 \xrightarrow{\phi_2} (H_{21} \otimes K_1) \oplus (H_{22} \otimes K_2) \oplus (H_{23} \otimes K_3) \end{pmatrix}$$

(b) The 2-morphism ϕ as a matrix of linear maps.

$$\phi_{i,j} : J_i \to H_{i,j} \otimes K_j \quad \text{for } i \in [2] \text{ and } j \in [3]$$

(c) The 2-morphism ϕ as a family of linear maps indexed by its adjacent regions.



(d) Graphical representation of the 2-morphism ϕ.

Figure 6 Translating between equivalent expressions for 2-morphisms.

Composition of 1-morphisms is given by 'matrix multiplication' of matrices of Hilbert spaces, with addition and multiplication of complex numbers replaced by direct sum and tensor product, respectively. Composition of 2-morphisms is given by componentwise composition of linear maps.

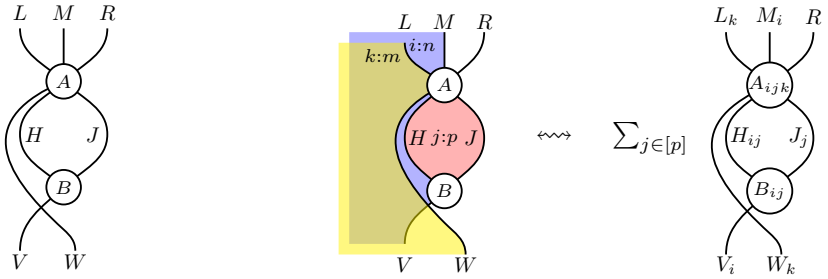
In the graphical calculus, regions, wires and vertices represent objects, 1-morphisms and 2-morphisms, respectively. The 2-category has a monoidal structure, acting on objects as multiplication, and on 1- and 2-morphisms as the Kronecker product of matrices of Hilbert spaces and linear maps, respectively; this is represented graphically by 'layering' one diagram above another.

Elementary description. While these structures are now well-understood in higher representation theory, they are not yet prevalent in the computer science community. To help the reader understand these new concepts, we also give a direct account of the formalism in elementary terms, that can be used without reference to the higher categorical technology. In Figure 6 we indicate how to translate between the categorical language presented above and the more elementary language used here.

In this direct perspective, shaded regions are labelled by *finite sets*, indexed by a parameter; we write $i{:}n$ to indicate that the parameter i varies over the set $[n]$.[9] Wires and vertices now represent *families* of Hilbert spaces and linear maps respectively, indexed by the parameters of all adjoining regions. A composite surface diagram represents a family of

[8] Here and throughout, we use the term '2-category' to refer to the fully weak structure, which is sometimes called 'bicategory'.

[9] For simplicity we will often omit these labels.



(a) An ordinary tensor diagram. **(b)** A shaded tensor diagram.

Figure 7 The graphical calculus.

composite linear maps, indexed by the parameters of all open regions, with closed regions being summed over.

We give an example in Figure 7(b). The coloured diagram on the left represents an entire family of composite linear maps. The maps which comprise this family are given by the right-hand diagrams for different values of k and i, which index the open regions. The closed region labelled $j : p$ is summed over.

Given this interpretation of diagrams D as families of linear maps D_i, we define two diagrams D, D' to be equal when all the corresponding linear maps D_i, D'_i are equal, and the scalar product λD as the family of linear maps λD_i.

Duality. We define the linear maps $\eta : \mathbb{C} \to \mathbb{C}^n \otimes \mathbb{C}^n$ and $\epsilon : \mathbb{C}^n \otimes \mathbb{C}^n \to \mathbb{C}$ as follows:

$$\eta : 1 \mapsto \sum_i |i\rangle \otimes |i\rangle \qquad \epsilon : |i\rangle \otimes |j\rangle \mapsto \delta_{ij} \tag{2}$$

Assuming for simplicity that all Hilbert spaces are chosen to be of the form $\mathbb{C}^n$ for some $n \in \mathbb{N}$, we introduce the following notation for families of linear maps of the form (2):

(3)

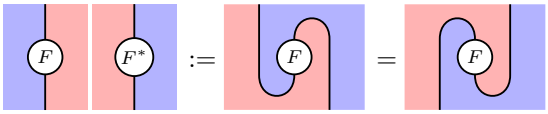
The notation is justified, since the following equations can be demonstrated:

(4)

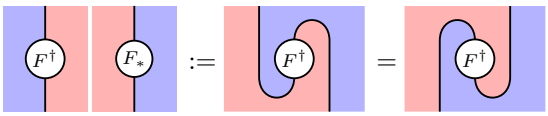
Dagger structure. Given a family of linear maps, we define its *adjoint* (or *dagger*) to be the family consisting of the adjoints of the linear maps. Graphically, we can think of the adjoint as a reflection about a horizontal axis. This is justified, since the following holds:

(5)

In total, every vertex appears in four variants:



(6)



(7)

The equations on the right-hand sides can be shown to follow from the definitions (2).

A dagger structure gives rise to a general notion of unitarity.

▶ **Definition 1.** A vertex U is *unitary* when it satisfies the following equations:

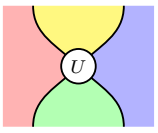
(8)

Standard boundaries. We only make use of a restricted portion of this calculus: wires which bound only one shaded region correspond to the 1-dimensional Hilbert space $\mathbb{C}$ for any value of the controlling parameter. (Hilbert spaces that do not bound regions may be of any finite dimension.) In particular, since they are 1-dimensional, the Hilbert spaces arising from these boundaries are not depicted in the corresponding family of tensor diagrams:

(9)

Biunitaries. Having defined our graphical calculus, we introduce our main algebraic entity.

▶ **Definition 2.** A *biunitary* is a vertex



(10)

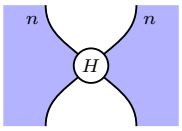
which satisfies the equations of Figure 1 for some scalar $\lambda \in \mathbb{C}$.

Note that biunitarity depends on a chosen partition of the input and output wires into two parts, which is extra data. Also, the scalar λ is uniquely determined by the type of U, and necessarily real and positive.

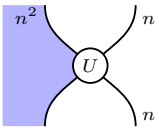
We will usually use the following equivalent formulation of biunitarity.

▶ **Definition 3.** The *clockwise* and *anticlockwise quarter rotation* of a vertex U of type (10) is given by the following composites, respectively:

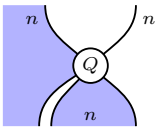
(11)



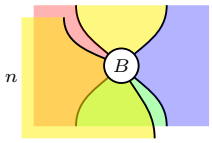
(a) Hadamard matrix.



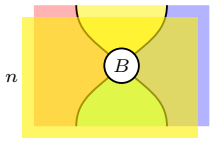
(b) Unitary error basis.



(c) Quantum Latin square.



(d) Half-plane controlled family.



(e) Full-plane controlled family.

Figure 8 Quantum structures and their corresponding biunitary types.

▶ **Proposition 4.** *Given a vertex U of type* (10)*, the following are equivalent:*
1. *U is biunitary;*
2. *U is unitary, and its clockwise quarter rotation is proportional to a unitary;*
3. *U is unitary, and its anticlockwise quarter rotation is proportional to a unitary.*
Furthermore, in cases 2 and 3, the proportionality factor is unique up to a phase and given by a square root of λ.

▶ **Corollary 5.** *Given a biunitary, arbitrary quarter rotations, or reflections about horizontal or vertical axes, are again proportional to biunitaries.*

In particular, as soon as we have characterized specific quantum structures in terms of biunitaries of certain types, we know that rotated and reflected versions of this type also correspond to this quantum structure, possibly after multiplication by a scalar.

2.2 Characterizing quantum structures

In this section we recall the biunitary characterizations of Hadamard matrices and unitary error bases, and give new characterizations of quantum Latin squares and controlled families. These results are summarized in Figure 8. In the following diagrams, all wires are either standard boundaries (9) or Hilbert spaces that do not bound any region.

Dimensional constraints. For a linear map $f : H \to J$ to be unitary imposes certain algebraic constraints on the dimensions of H and J; namely, $\dim(H) = \dim(J)$. For a vertex of type (10) to be biunitary similarly induces certain constraints on the allowed labels for the surrounding regions and wires. In all cases, these constraints are easily identified and solved for. For example, consider the following vertex U and its clockwise quarter rotation:

(12)

Here, n, m and p denote the dimensions of the corresponding region or wire, respectively. For the first of these to be unitary requires that $n = m$, while for the second to be unitary requires $p = nm$. By Theorem 4, for U to be biunitary, we therefore require $(n, m, p) = (n, n, n^2)$, and the set of allowed types is parameterized by a single natural number. For the rest of this section, we will label biunitaries by their allowed dimensions.

Hadamard matrices. Hadamard matrices were identified by Jones to be characterized in terms of biunitarity [23]. Complex Hadamard matrices play an important role in mathematical physics and quantum information theory [18]; in particular, they encode the data of a basis which is unbiased with respect to the computational basis.

▶ **Definition 6.** A *Hadamard matrix* is a matrix $H \in \mathrm{Mat}_n(\mathbb{C})$ with the following properties, for $i, j \in [n]$:

$$H_{i,j}\overline{H}_{i,j} = 1 \sum_k H_{i,k}\overline{H}_{j,k} = \delta_{i,j} n \qquad \sum_k \overline{H}_{k,i} H_{k,j} = \delta_{i,j} n \tag{13}$$

The last two equations are equivalent, but we include them both for completeness.

The biunitary characterization of Hadamard matrices is due to Jones in the setting of the spin model planar algebra, which our mathematical setup generalizes. It was shown in [53, Theorem 4.5] that this characterization is equivalent to that of Coecke and Duncan in terms of interacting Frobenius algebras [12].

▶ **Proposition 7** (Jones [23, Section 2.11])**.** *Hadamard matrices of dimension n correspond to biunitaries of the type shown in Figure 8(a).*

Unitary error bases. Originally introduced by Knill [31], unitary error bases are ubiquitous in modern quantum information theory. They lie at the heart of quantum error correction [49] and procedures such as superdense coding and quantum teleportation [55].

▶ **Definition 8** (Knill [31])**.** A *unitary error basis* (UEB) on an n-dimensional Hilbert space H is a collection of unitary matrices $\{U_a \in \mathrm{U}(H) \,|\, a \in [n^2]\}$, satisfying the following orthogonality property, for $a, b \in [n^2]$:

$$\mathrm{Tr}(U_a^\dagger U_b) = n\, \delta_{a,b} \tag{14}$$

That is, a UEB is an orthogonal basis of the space $\mathrm{End}(H)$ consisting of unitary matrices.

We denote the (i, j)th matrix element of the matrix U_a by $U_{a,i,j} = (U_a)_{i,j} = \langle i|\, U_a \,|j\rangle$.

▶ **Proposition 9** (Vicary [53, Theorem 4.2])**.** *Unitary error bases on an n-dimensional Hilbert space correspond to biunitaries of the type shown in Figure 8(b).*

Quantum Latin squares. Quantum Latin squares were introduced by Musto and the second author [36] as generalizations of classical Latin squares, with applications to the construction of unitary error bases. Related constructions were also introduced independently by Banica and Nicoară [7].

▶ **Definition 10** (Musto & V. [36, Definition 1])**.** A *quantum Latin square* (QLS) on an n-dimensional Hilbert space H is a square grid of vectors $\{|Q_{a,b}\rangle \in H \,|\, a, b \in [n]\}$ such that each row $\{|Q_{a,b}\rangle \,|\, b \in [n]\}$ and each column $\{|Q_{a,b}\rangle \,|\, a \in [n]\}$ form an orthonormal basis of H; for $a, b, c \in [n]$:

$$\langle Q_{a,b}|Q_{a,c}\rangle = \delta_{b,c} \qquad \langle Q_{a,c}|Q_{b,c}\rangle = \delta_{a,b} \tag{15}$$

We denote the ith entry of the vector $|Q_{a,b}\rangle$ by $Q_{a,b,i} = \langle i|Q_{a,b}\rangle$.

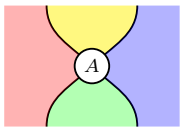
▶ **Proposition 11.** *Quantum Latin squares on an n-dimensional Hilbert space correspond to biunitaries of the type shown in Figure 8(c).*

Controlled families. In quantum information we often want to describe lists of structures, parameterized by a given index. A standard name for such a list is a controlled family.

▶ **Definition 12.** For a given quantum structure X, an *n-controlled family* is an ordered list of n instances of X.

In index notation, we reserve superscript for controlling indices. For example, a controlled family of Hadamard matrices would be written as $H^c_{a,b}$, where c iterates through the controlled family and a and b are the actual indices of the Hadamard matrix H^c.

▶ **Proposition 13.** *An n-controlled family of biunitaries of type*



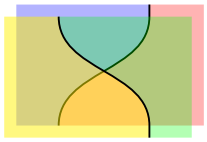
(16)

corresponds to a biunitary of the same type with an additional half-plane or full-plane sheet of dimension n attached, as shown in Figure 8(d) and (e).

By Corollary 5, we could have put the half-plane controlling sheet in one of 4 different orientations. Furthermore, it makes no difference if the controlling sheet goes in front or behind. We therefore have 8 different half-plane controls and 2 different full plane controls. For example. the type shown in Figure 2(d) indicates a controlled family of Hadamards.

In our pseudo-3d graphical notation, it can be hard to see if a rear sheet is actually connected to a vertex. In our diagrams, we will use the convention that all sheets drawn beneath a vertex are connected to it.

Interchangers. The vertex representing the crossing of wires at different depths is called an *interchanger*:



(17)

This is given canonically for all index values as the swap map $H \otimes J \to J \otimes H$.

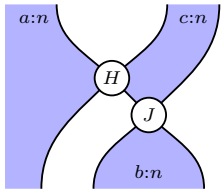
▶ **Proposition 14.** *The interchanger* (17) *is biunitary.*

3 Biunitary composition

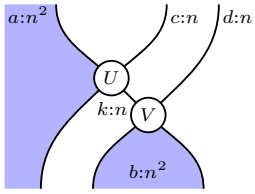
The results of this section are corollaries of the following idea.

▶ **Theorem 15.** *Arbitrary finite diagonal composites of biunitaries are again biunitary.*

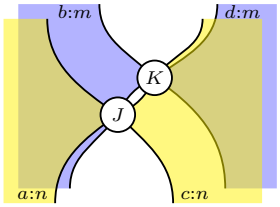
Since we have established in Section 2 that biunitaries of various types correspond to different quantum structures, Theorem 15 suggests the possibility of building new quantum structures from existing ones by diagonal composition. In Section 3.1, we demonstrate that binary diagonal composites of biunitaries are again biunitary. We then consider the problem of diagonally composing the biunitaries corresponding to Hadamards, quantum Latin squares, unitary error bases and controlled families to produce other such structures, investigating binary composites in Section 3.2, ternary composites in Section 3.3, and higher composites in Section 3.4. In an extended version of this paper [44], we argue that our methods gives rise to an infinite number of genuinely distinct constructions.



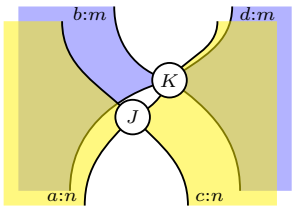
(a) $\mathrm{Had}_n + \mathrm{Had}_n \rightsquigarrow \mathrm{QLS}_n$



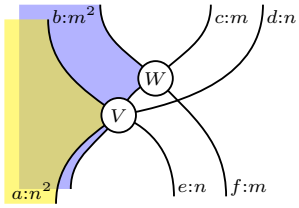
(b) $\mathrm{UEB}_n + \mathrm{UEB}_n \rightsquigarrow \mathrm{QLS}_{n^2}$



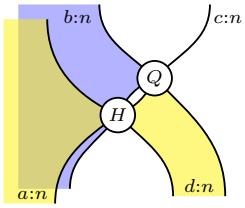
(c) $\mathrm{Had}_n^m + \mathrm{Had}_m^n \rightsquigarrow \mathrm{Had}_{nm}$ (Hosoya-Suzuki)



(d) $\mathrm{Had}_n + \mathrm{Had}_m^n \rightsquigarrow \mathrm{Had}_{nm}$



(e) $\mathrm{UEB}_n^{m^2} + \mathrm{UEB}_m \rightsquigarrow \mathrm{UEB}_{nm}$



(f) $\mathrm{Had}_n^n + \mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_n$

Figure 9 Binary biunitary constructions.

3.1 Diagonal composition

It is straightforward to see that the diagonal composite of two biunitaries is again biunitary.

▶ **Theorem 16.** *Let U, V and W be biunitaries with the types illustrated below.*

Then the following diagonal composites are biunitary, with respect to the indicated partitions of the input and output wires:

(18)

Except for the pinwheel composite[10] [16], which can be handled separately, this shows that Theorem 15 holds.

3.2 Binary composites

We give a number of quantum constructions in Figure 9, each involving the diagonal composite of two biunitaries. Correctness of all these constructions follows as corollaries from Theorem 16, and the results of Section 2.2 as summarized in Figure 8.

Quantum Latin squares. We begin by presenting two quantum Latin square constructions. The following construction produces a quantum Latin square from two Hadamard matrices, generalizing [7, Definition 2.3] and [36, Definition 10].

[10] The *pinwheel composite* is a way to compose five 2-morphisms in a double category, in a way which cannot be described in terms of repeated binary composites.

▶ **Corollary 17** ($\mathrm{Had}_n + \mathrm{Had}_n \rightsquigarrow \mathrm{QLS}_n$)**.** *The construction of Figure 9(a) produces an n-dimensional quantum Latin square*

$$Q_{a,b,c} = \tfrac{1}{\sqrt{n}}\, H_{a,c}\, J_{c,b} \tag{19}$$

from the following data, with $a, b, c \in [n]$:

- *$H_{a,c}$ and $J_{c,b} \in \mathrm{Had}_n$, n-dimensional Hadamards.*

The factor $\frac{1}{\sqrt{n}}$ arises as described in Theorem 4, since the biunitary J is of rotated Hadamard type. Such a biunitary is a *unitary* matrix; given an ordinary Hadamard matrix, we need to rescale it by a factor of $\frac{1}{\sqrt{n}}$ to obtain such a unitary. A similar comment applies to several of the constructions below. The next construction, which we believe to be new, produces a quantum Latin square from two unitary error bases.

▶ **Corollary 18** ($\mathrm{UEB}_n + \mathrm{UEB}_n \rightsquigarrow \mathrm{QLS}_{n^2}$)**.** *The construction of Figure 9(b) produces an n^2-dimensional quantum Latin square*

$$Q_{a,b,cd} = \tfrac{1}{\sqrt{n}} \textstyle\sum_{k\in[n]} U_{a,c,k}\, V_{b,k,d} \tag{20}$$

from the following data, with $a, b \in [n^2]$ and $c, d \in [n]$:

- *$U_{a,c,k}$ and $V_{b,k,d} \in \mathrm{UEB}_n$, n-dimensional UEBs.*

Note that we concatenate indices corresponding to tensor products of Hilbert spaces or products of indexing sets; for example, for a QLS on a Hilbert space $V \otimes W$, the coefficient of the basis vector $|i, j\rangle = |i\rangle \otimes |j\rangle$ in the (a, b)th position of the quantum Latin square will be written as $Q_{a,b,ij}$. Similarly, if the indexing set of a UEB is the product of two sets $[n] \times [m]$ we denote its (a, b)th element by U_{ab} with coefficients $U_{ab,i,j}$.

Hadamard matrices. The following construction produces a single Hadamard matrix from two controlled families.

▶ **Corollary 19** ($\mathrm{Had}_n^m + \mathrm{Had}_m^n \rightsquigarrow \mathrm{Had}_{nm}$)**.** *The construction of Figure 9(c) produces an nm-dimensional Hadamard matrix*

$$H_{ab,cd} = J^b_{a,c}\, K^c_{b,d} \tag{21}$$

from the following data, with $a, c \in [n]$ and $b, d \in [m]$:

- *$J^b_{a,c} \in \mathrm{Had}_n^m$, an m-controlled family of n-dimensional Hadamard matrices;*
- *$K^c_{b,d} \in \mathrm{Had}_m^n$, an n-controlled family of m-dimensional Hadamard matrices.*

This construction was introduced in 2003 by Hosoya and Suzuki [21] under the name *generalized tensor product*. A better known special case, due to Diţă [17], is a central tool in the study and classification of Hadamard matrices; we give it explicitly in Figure 9(d).

Unitary error bases. We now turn our attention to unitary error bases. By a manual combinatorial check, it can be verified that the constructions in Figure 9(e) and (f) are the only possible binary constructions of UEBs using only Hadamard matrices, UEBs or QLSs and controlled families thereof.

▶ **Corollary 20** ($\mathrm{UEB}_n^{m^2} + \mathrm{UEB}_m \rightsquigarrow \mathrm{UEB}_{nm}$)**.** *The construction of Figure 9(e) produces an nm-dimensional unitary error basis*

$$U_{ab,cd,ef} = V^b_{a,d,e}\, W_{b,c,f} \tag{22}$$

from the following data, with $a \in [n^2]$, $b \in [m^2]$, $c, f \in [m]$ and $d, e \in [n]$:

- $V^b_{a,d,e} \in \mathrm{UEB}^{m^2}_n$, *an* m^2*-controlled family of* n*-dimensional unitary error bases;*
- $W_{b,c,f} \in \mathrm{UEB}_m$, *an* m*-dimensional unitary error basis.*

In Figure 9(e), we have used biunitarity of the interchanger as established in Proposition 14.

It is also possible to compose biunitaries of different types to obtain unitary error bases, as shown by the following biunitary characterization of an existing construction, the *quantum shift-and-multiply* method [36], which simultaneously generalizes the shift-and-multiply method [55] and the Hadamard method [36, Definition 33].

▶ **Corollary 21** ($\mathrm{Had}^n_n + \mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_n$)**.** *The construction of Figure 9(f) produces an* n*-dimensional unitary error basis*

$$U_{ab,c,d} = H^b_{a,d}\, Q_{b,d,c} \tag{23}$$

from the following data, with $a, b, c, d \in [n]$*:*

- $H^b_{a,d} \in \mathrm{Had}^n_n$, *an* n*-controlled family of* n*-dimensional Hadamard matrices;*
- $Q_{b,d,c} \in \mathrm{QLS}_n$, *an* n*-dimensional quantum Latin square.*

3.3 Ternary constructions

Here we list all ternary biunitary constructions of unitary error bases from Hadamard matrices, unitary error bases, quantum Latin squares and controlled families thereof, which do not factor through constructions of lower arity. We summarize them in Figure 10(a)–(d). Up to equivalence, we assert that this list is complete, although we do not prove completeness in a formal way. To our knowledge, all constructions in this section are new. As before, all these results are corollaries of Theorem 16, and the results of Section 2.2 as summarized in Figure 8.

The constructions of Figure 10(a) and Figure 10(b) can be seen as slight alterations of constructions that factor through the constructions of Figure 9.

▶ **Corollary 22** ($\mathrm{Had}^{m^2,n}_n + \mathrm{UEB}^{n,n}_m + \mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_{nm}$)**.** *The construction of Figure 10(a) produces an* nm*-dimensional UEB*

$$U_{abc,de,fg} := H^{b,c}_{a,f}\, V^{c,f}_{b,e,g}\, Q_{c,f,d} \tag{24}$$

from the following data, with $a, c, d, f \in [n]$*,* $b \in [m^2]$ *and* $e, g \in [m]$*:*

- $H^{b,c}_{a,f} \in \mathrm{Had}^{m^2,n}_n$, *an* (m^2, n)*-controlled family of* n*-dimensional Hadamard matrices;*
- $V^{c,f}_{b,e,g} \in \mathrm{UEB}^{n,n}_m$, *an* (n, n)*-controlled family of* m*-dimensional unitary error bases;*
- $Q_{c,f,d} \in \mathrm{QLS}_n$, *an* n*-dimensional quantum Latin square.*

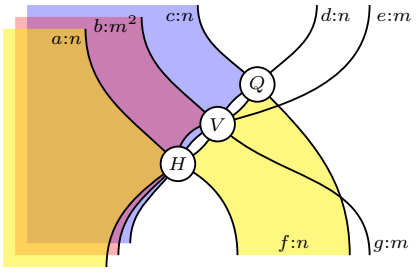
▶ **Corollary 23** ($\mathrm{Had}^{n,m}_{nm} + \mathrm{QLS}^{m,m}_n + \mathrm{QLS}_m \rightsquigarrow \mathrm{UEB}_{nm}$)**.** *The construction of Figure 10(b) produces an* nm*-dimensional UEB*

$$U_{abc,de,fg} := H^{b,c}_{a,eg}\, P^{c,g}_{e,b,f}\, Q_{c,g,d} \tag{25}$$

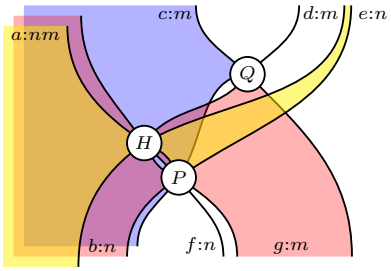
from the following data, with $a \in [nm]$ $b, e, f \in [n]$ *and* $c, d, g \in [m]$*:*

- $H^{b,c}_{a,eg} \in \mathrm{Had}^{n,m}_{nm}$, *an* (n, m)*-controlled family of* nm*-dimensional Hadamard matrices;*
- $P^{c,g}_{e,b,f} \in \mathrm{QLS}^{m,m}_n$, *an* (m, m)*-controlled family of* n*-dimensional quantum Latin squares;*
- $Q_{c,g,d} \in \mathrm{QLS}_m$, *an* m*-dimensional quantum Latin square.*

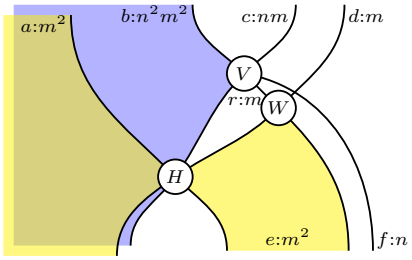
The following is geometrically the simplest of our ternary constructions. It involves a closed wire, so the index expression includes a sum.



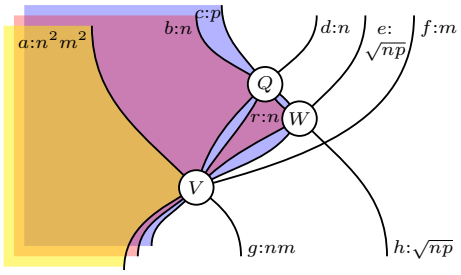
(a) $\mathrm{Had}_n^{m^2,n} + \mathrm{UEB}_m^{n,n} + \mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_{nm}$



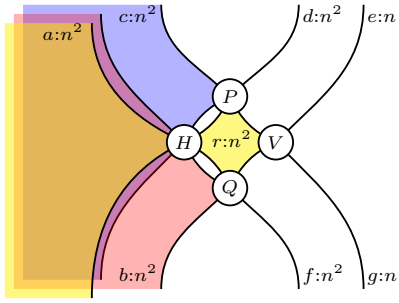
(b) $\mathrm{Had}_{nm}^{n,m} + \mathrm{QLS}_n^{m,m} + \mathrm{QLS}_m \rightsquigarrow \mathrm{UEB}_{nm}$



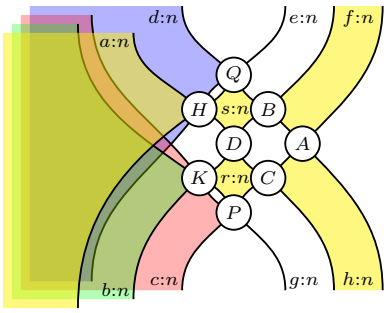
(c) $\mathrm{Had}_{m^2}^{n^2m^2} + \mathrm{UEB}_{nm} + \mathrm{UEB}_m \rightsquigarrow \mathrm{UEB}_{nm^2}$



(d) $\mathrm{UEB}_{nm}^{n,p} + \mathrm{QLS}_n^p + \mathrm{UEB}_{\sqrt{np}} \rightsquigarrow \mathrm{UEB}_{nm\sqrt{np}}$



(e) $\mathrm{Had}_{n^2}^{n^2,n^2} + 2{\times}\mathrm{QLS}_{n^2} + \mathrm{UEB}_n \rightsquigarrow \mathrm{UEB}_{n^3}$



(f) $4{\times}\mathrm{Had}_n + 2{\times}\mathrm{Had}_n^n + 2{\times}\mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_{n^2}$

Figure 10 Higher-order unitary error basis constructions.

▶ **Corollary 24** ($\mathrm{Had}_{m^2}^{n^2m^2} + \mathrm{UEB}_{nm} + \mathrm{UEB}_m \rightsquigarrow \mathrm{UEB}_{nm^2}$)**.** *The construction of Figure 10(c) produces an nm^2-dimensional UEB*

$$U_{ab,cd,ef} := \sum_{r\in[m]} H^b_{a,e}\, V_{b,c,rf}\, W_{e,r,d} \tag{26}$$

from the following data, with $a, e \in [m^2]$, $b \in [n^2m^2]$, $c \in [nm]$, $d \in [m]$, and $f \in [n]$:

- $H^b_{a,e} \in \mathrm{Had}_{m^2}^{n^2m^2}$*, an n^2m^2-controlled family of m^2-dimensional Hadamard matrices;*
- $V_{b,c,rf} \in \mathrm{UEB}_{nm}$*, an nm-dimensional unitary error basis;*
- $W_{e,r,d} \in \mathrm{UEB}_m$*, an m-dimensional unitary error basis.*

Our final ternary construction is the first to involve a sum over a closed region, which again gives rise to a summation.

▶ **Corollary 25** ($\mathrm{UEB}_{nm}^{n,p} + \mathrm{QLS}_n^p + \mathrm{UEB}_{\sqrt{np}} \rightsquigarrow \mathrm{UEB}_{nm\sqrt{np}}$)**.** *For $n, m, p \in \mathbb{N}$ such that $\sqrt{np} \in \mathbb{N}$, the construction of Figure 10(d) produces an $nm\sqrt{np}$-dimensional UEB*

$$U_{abc,def,gh} := \sum_{r\in[n]} V^{b,c}_{a,rf,g}\, Q^c_{b,r,d}\, W_{rc,e,h} \tag{27}$$

from the following data, with $a \in [n^2m^2]$, $b, d \in [n], c \in [p]$, $e, h \in [\sqrt{np}]$, $f \in [m]$, *and* $g \in [nm]$*:*

- $V^{b,c}_{a,rf,g} \in \mathrm{UEB}^{n,p}_{nm}$, *an* (n,p)*-controlled family of* nm*-dimensional unitary error bases;*
- $Q^{c}_{b,r,d} \in \mathrm{QLS}^{p}_{n}$, *an* p*-controlled family of* n*-dimensional quantum Latin squares;*
- $W_{rc,e,h} \in \mathrm{UEB}_{\sqrt{np}}$, *an* $\sqrt{np}$*-dimensional unitary error basis.*

3.4 Higher constructions

Interesting biunitary composites exist at higher arity, and are easy to find by experimentation. We give two examples which seem elegant, and which we believe are new.

▶ **Corollary 26** ($\mathrm{Had}^{n^2,n^2}_{n^2} + 2{\times}\mathrm{QLS}_{n^2} + \mathrm{UEB}_n \rightsquigarrow \mathrm{UEB}_{n^3}$)**.** *The construction in Figure 10(e) produces an* n^3*-dimensional UEB*

$$U_{abc,de,fg} = \textstyle\sum_{r\in[n^2]} H^{b,c}_{a,r}\, P_{c,r,d}\, Q_{r,b,f}\, V_{r,e,g} \tag{28}$$

from the following data, with $a, b, c, d, f \in [n^2]$ *and* $e, g \in [n]$*:*

- $H^{b,c}_{a,r} \in \mathrm{Had}^{n^2,n^2}_{n^2}$, *an* (n^2, n^2)*-controlled family of* n^2*-dimensional Hadamard matrices;*
- $P_{c,r,d}, Q_{r,b,f} \in \mathrm{QLS}_{n^2}$, n^2*-dimensional quantum Latin squares;*
- $V_{r,e,g} \in \mathrm{UEB}_n$, *an* n*-dimensional unitary error bases.*

In an extended version of this paper [44], we use this construction to produce a new unitary error basis that cannot be obtained by the most general previously known methods.

Finally, we give the 8-ary construction of Figure 10(f).

▶ **Corollary 27** ($4{\times}\mathrm{Had}_n + 2{\times}\mathrm{Had}^{n}_{n} + 2{\times}\mathrm{QLS}_n \rightsquigarrow \mathrm{UEB}_{n^2}$)**.** *The construction in Figure 10(f) produces an* n^2*-dimensional UEB*

$$U_{abcd,ef,gh} = \tfrac{1}{n} \textstyle\sum_{r,s\in[n]} A_{f,h}\, B_{s,f}\, C_{r,h}\, D_{s,r}\, H^{d}_{a,s}\, K^{c}_{b,r}\, Q_{d,s,e}\, P_{r,c,g} \tag{29}$$

from the following data, with $a, b, c, d, e, f, g, h \in [n]$*:*

- $A_{f,h}, B_{s,f}, C_{r,h}, D_{s,r} \in \mathrm{Had}_n$, n*-dimensional Hadamard matrices;*
- $H^{d}_{a,s}, K^{c}_{b,r} \in \mathrm{Had}^{n}_{n}$, n*-controlled families of* n*-dimensional Hadamard matrices;*
- $Q_{d,s,e}, P_{r,c,g} \in \mathrm{QLS}_n$, n*-dimensional quantum Latin squares.*

Acknowledgements. We are grateful to Bruce Bartlett, André Henriques, Scott Morrison, Benjamin Musto and Dominic Verdon for useful discussions. We also thank the anonymous referees for valuable suggestions. Early versions of these results were presented at WIP 2016 and QIP 2017, and we are grateful to the organizers of these events.

References

1 Samson Abramsky and Bob Coecke. A categorical semantics of quantum protocols. In *Proceedings of LICS*, 2004. doi:10.1109/lics.2004.1319636.

2 Samson Abramsky and Bob Coecke. Categorical quantum mechanics. In *Handbook of Quantum Logic and Quantum Structures*, pages 261–323. Elsevier, 2009. doi:10.1016/B978-0-444-52869-8.50010-4.

3 Miriam Backens. The ZX-calculus is complete for stabilizer quantum mechanics. *NJP*, 16:093021, 2014. doi:10.1088/1367-2630/16/9/093021.

4 John C. Baez. Higher-dimensional algebra II. 2-Hilbert spaces. *Advances in Mathematics*, 127(2):125–189, 1997. doi:10.1006/aima.1997.1617.

5 Teodor Banica, Julien Bichon, and Benoît Collins. Quantum permutation groups: a survey. In *Noncommutative Harmonic Analysis with Applications to Probability*. Polish Academy of Sciences, 2007. doi:10.4064/bc78-0-1.

6 Teodor Banica, Julien Bichon, and Jean-Marc Schlenker. Representations of quantum permutation algebras. *Journal of Functional Analysis*, 257(9):2864–2910, 2009. doi:10.1016/j.jfa.2009.04.013.

7 Teodor Banica and Remus Nicoară. Quantum groups and Hadamard matrices. *Panamerican Mathematical Journal*, 17:1–24, 2007. arXiv:math/0610529.

8 John W. Barrett, Catherine Meusburger, and Gregor Schaumann. Gray categories with duals and their diagrams. *to appear*, 2012. arXiv:1211.0529.

9 Bruce Bartlett. Quasistrict symmetric monoidal 2-categories via wire diagrams, 2014. arXiv:1409.2148.

10 Tristan Benoist and Ion Nechita. On bipartite unitary matrices generating subalgebra-preserving quantum operations. *Linear Algebra and its Applications*, 521:70–103, 2017. doi:10.1016/j.laa.2017.01.020.

11 Bob Coecke. Kindergarten quantum mechanics: Lecture notes. In *AIP Conference Proceedings*. AIP Publishing, 2006. doi:10.1063/1.2158713.

12 Bob Coecke and Ross Duncan. Interacting quantum observables. In *Automata, Languages and Programming*, volume 5126, pages 298–310. Springer, 2008. doi:10.1007/978-3-540-70583-3_25.

13 Bob Coecke, Chris Heunen, and Aleks Kissinger. Categories of quantum and classical channels. *Quant. Inf. Proc.*, 2014. doi:10.1007/s11128-014-0837-4.

14 Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes*. Cambridge University Press, 2017.

15 Bob Coecke, Dusko Pavlovic, and Jamie Vicary. A new description of orthogonal bases. *Mathematical Structures in Computer Science*, 23(03):555–567, 2012. doi:10.1017/s0960129512000047.

16 Robert Dawson and Robert Paré. Characterizing tileorders. *Order*, 10(2):111–128, 1993. doi:10.1007/bf01111295.

17 Petre Diţă. Some results on the parametrization of complex Hadamard matrices. *Journal of Physics A*, 37(20):5355–5374, 2004. doi:10.1088/0305-4470/37/20/008.

18 Thomas Durt, Berthold-Georg Englert, Ingemar Bengtsson, and Karol Życzkowski. On mutually unbiased bases. *International Journal of Quantum Information*, 08(04):535–640, 2010. doi:10.1142/s0219749910006502.

19 Josep Elgueta. A strict totally coordinatized version of Kapranov and Voevodsky's 2-category 2Vect. *Mathematical Proceedings of the Cambridge Philosophical Society*, 142(03):407, 2007. doi:10.1017/s0305004106009881.

20 Uffe Haagerup. Orthogonal maximal abelian $*$-subalgebras of the n×n matrices and cyclic n-roots. *Institut for Matematik*, 29:296–322, 1996.

21 Rie Hosoya and Hiroshi Suzuki. Type II matrices and their Bose-Mesner algebras. *Journal of Algebraic Combinatorics*, 17:19–37, 2003. doi:10.1023/a:1021960623533.

22 Benjamin Hummon. *Surface diagrams for Gray categories*. PhD thesis, UC San Diego, 2012. URL: http://escholarship.org/uc/item/5b24s9cc.

23 Vaughan F. R. Jones. Planar algebras, I, 1999. arXiv:math/9909027.

24 Vaughan F. R. Jones, Scott Morrison, and Noah Snyder. The classification of subfactors of index at most 5. *Bull. Amer. Math. Soc.*, 51(2):277–327, 2013. doi:10.1090/s0273-0979-2013-01442-3.

25 Vaughan F.R. Jones. On knot invariants related to some statistical mechanical models. *Pacific Journal of Mathematics*, 137:311–334, 1989. doi:10.2140/pjm.1989.137.311.

26 Vaughan F.R. Jones and Vaikalathur S. Sunder. *Introduction to Subfactors*. Cambridge University Press (CUP), 1997. doi:10.1017/cbo9780511566219.

27 André Joyal and Ross Street. The geometry of tensor calculus, I. *Advances in Mathematics*, 88(1):55–112, 1991. doi:10.1016/0001-8708(91)90003-p.

28 Aleks Kissinger and Vladimir Zamdzhiev. Quantomatic: A proof assistant for diagrammatic reasoning. In *Automated Deduction - CADE-25*, pages 326–336. Springer International Publishing, 2015. doi:10.1007/978-3-319-21401-6_22.

29 Andreas Klappenecker and Martin Rötteler. Unitary error bases: Constructions, equivalence, and applications. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 139–149. Springer, 2003. doi:10.1007/3-540-44828-4_16.

30 Emanuel Knill. Group representations, error bases and quantum codes. *Los Alamos NLR LAUR-96-2807*, 1996. doi:10.2172/378680.

31 Emanuel Knill. Non-binary unitary error bases and quantum codes. *Los Alamos National Laboratory Report LAUR-96-2717*, 1996. doi:10.2172/373768.

32 Máté Matolcsi, Júlia Réffy, and Ferenc Szöllősi. Constructions of complex Hadamard matrices via tiling abelian groups. *Open Systems & Information Dynamics*, 14(03):247–263, 2007. doi:10.1007/s11080-007-9050-6.

33 David A. Meyer and Thomas G. Wong. Connectivity is a poor indicator of fast quantum search. *Physical Review Letters*, 114(11), 2015. doi:10.1103/physrevlett.114.110503.

34 Scott Morrison and Emily Peters. The little desert? Some subfactors with index in the interval $(5, 3 + \sqrt{5})$. *International Journal of Mathematics*, 25(08):1450080, 2014. doi:10.1142/s0129167x14500803.

35 Benjamin Musto. Constructing mutually unbiased bases from quantum Latin squares, 2016. arXiv:1605.08919.

36 Benjamin Musto and Jamie Vicary. Quantum Latin squares and unitary error bases. *Quantum Information and Computation*, 2016. to appear. arXiv:1504.02715.

37 Remus Nicoară. A finiteness result for commuting squares of matrix algebras. *Journal of Operator Theory*, 55(2):295–310, 2006. arXiv:math/0404301.

38 Remus Nicoară. Subfactors and Hadamard matrices. *Journal of Operator Theory*, 64(2):453–468, 2010. arXiv:0704.1128.

39 Adrian Ocneanu. Quantized groups, string algebras, and Galois theory for algebras. In *Operator Algebras and Applications*, pages 119–172. CUP, 1989. doi:10.1017/cbo9780511662287.008.

40 Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014. doi:10.1016/j.aop.2014.06.013.

41 Roger Penrose. Applications of negative-dimensional tensors. In D.J.A. Welsh, editor, *Combinatorial Mathematics and its Applications*, pages 221–244. Academic Press, New York, 1971.

42 Mihai Petrescu. *Existence of Continuous Families of Complex Hadamard Matrices of Certain Prime Dimensions and Related Results*. PhD thesis, University of California, Los Angeles, 1997.

43 Sorin Popa. Orthogonal pairs of $*$-subalgebras in finite von Neumann algebras. *Journal of Operator Theory*, 9:253–268, 1983.

44 David Reutter and Jamie Vicary. Biunitary constructions in quantum information, 2016. arXiv:1609.07775.

45 Christopher Schommer-Pries. *The classification of two-dimensional extended topological field theories*. PhD thesis, Department of Mathematics, University of California, Berkeley, 2009. arXiv:1112.1000.

46 Peter Selinger. Dagger compact closed categories and completely positive maps. *Electronic Notes in Theoretical Computer Science*, 170:139–163, 2007. doi:10.1016/j.entcs.2006.12.018.

47 Peter Selinger. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pages 289–355. Springer, 2010. doi:10.1007/978-3-642-12821-9_4.

48 Claude E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949. doi:10.1002/j.1538-7305.1949.tb00928.x.

49 Peter W. Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 56–65. IEEE Computer Society Press, 1996. doi:10.1109/sfcs.1996.548464.

50 Ferenc Szöllősi. *Construction, classification and parametrization of complex Hadamard matrices*. PhD thesis, Central European University, Budapest, Hungary, 2011. arXiv:1110.5590.

51 Ferenc Szöllősi. Complex Hadamard matrices of order 6: a four-parameter family. *Journal of the London Mathematical Society*, 85(3):616–632, 2012. doi:10.1112/jlms/jdr052.

52 Wojciech Tadej and Karol Życzkowski. A concise guide to complex Hadamard matrices. *Open Systems & Information Dynamics*, 13(02):133–177, 2006. doi:10.1007/s11080-006-8220-2.

53 Jamie Vicary. Higher quantum theory, 2012. URL: https://arxiv.org/abs/1207.4563.

54 Jamie Vicary. Higher semantics of quantum protocols. In *Proceedings of LICS*, 2012. doi:10.1109/lics.2012.70.

55 Reinhard F. Werner. All teleportation and dense coding schemes. *Journal of Physics A*, 34(35):7081–7094, 2001. doi:10.1088/0305-4470/34/35/332.

56 Pawel Wocjan and Thomas Beth. New construction of mutually unbiased bases in square dimension, 2004. arXiv:quant-ph/0407081.

Uniform Interpolation in Coalgebraic Modal Logic*

Fatemeh Seifan[1], Lutz Schröder[2], and Dirk Pattinson[3]

1 **Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany**
2 **Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany**
3 **Australian National University, Acton, Australia**

Abstract

A logic has *uniform interpolation* if its formulas can be projected down to given subsignatures, preserving all logical consequences that do not mention the removed symbols; the weaker property of *(Craig) interpolation* allows the projected formula – the *interpolant* – to be different for each logical consequence of the original formula. These properties are of importance, e.g., in the modularization of logical theories. We study interpolation in the context of coalgebraic modal logics, i.e. modal logics axiomatized in rank 1, restricting for clarity to the case with finitely many modalities. Examples of such logics include the modal logics K and KD, neighbourhood logic and its monotone variant, finite-monoid-weighted logics, and coalition logic. We introduce a notion of one-step (uniform) interpolation, which refers only to a restricted logic without nesting of modalities, and show that a coalgebraic modal logic has uniform interpolation if it has one-step interpolation. Moreover, we identify preservation of finite surjective weak pullbacks as a sufficient, and in the monotone case necessary, condition for one-step interpolation. We thus prove or reprove uniform interpolation for most of the examples listed above.

1998 ACM Subject Classification F.4.1 Mathematical Logic – Modal logic, I.2.4 Knowledge Representation Formalisms and Methods – Modal logic, I.2.3 Deduction and Theorem Proving

Keywords and phrases Coalgebraic modal logic, uniform interpolation, weak pullback

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.21

1 Introduction

Given a logic with a notion of formula and signature (and featuring implication for simplicity), the *Craig interpolation* property requires that every valid implication $\phi \to \psi$ has an *interpolant*, i.e. a formula ρ mentioning only the signature symbols that occur in both ϕ and ψ, such that both $\phi \to \rho$ and $\rho \to \psi$ are valid. The stricter *uniform interpolation* property additionally demands that ρ can be made to depend only on ϕ and on the signature of ψ (or, yet stricter, on the shared symbols of ϕ and ψ), rather than also on ψ itself. Both Craig interpolation and uniform interpolation are useful in the structuring and modularization of logical theories for purposes of specification and automated deduction, e.g. in large ontologies [39, 19]. Craig interpolation was originally proved for first-order logic [5] and later extended to many other systems, notably various modal logics including the basic modal logic K [8], as well as intuitionistic logic [9] and the μ-calculus [6]. Uniform interpolation is easily seen to hold for classical propositional logic but in fact fails for first-order predicate logic [16]. Intuitionistic logic [30], the basic modal logic K [10, 38], and the modal μ-calculus [17] do have uniform interpolation, while it fails for the modal logics $S4$ [11] and $K4$ [3].

* Work by the first and second author forms part of DFG project GenMod3 (SCHR 1118/5-3)

In this paper, we study interpolation and uniform interpolation in the context of predicate-lifting style coalgebraic modal logic [26, 34], equivalently, of modal logics that are axiomatized by *rank*-1 axioms [33, 36]. Coalgebraic modal logic is a generic framework for modal logics whose semantics goes beyond the standard relational world, and e.g. includes probabilistic, game-based, neighbourhood-based, or weighted behaviour. It is parametrized over the choice of a *type functor* (in our setting, on the category of sets), whose coalgebras play the role of models. The name of the game in coalgebraic logic is to reduce properties of the full modal logic to properties of the *one-step* logic, which restricts to formulas with exactly one layer of modalities and is interpreted over very simple structures that essentially capture the collection of successors of a single state in a model. Following this paradigm, we identify a notion of one-step interpolation, and then establish that for a coalgebraic modal logic $\mathcal{L}$ with finitely many modalities, the following properties imply each other in sequence:

1. the modalities are *separating*, i.e. support a Hennessy-Milner-style expressivity theorem [26, 34] (implying that the type functor preserves finite sets), and the type functor preserves surjective finite weak pullbacks (which for finitary functors just means that the functor preserves surjective weak pullbacks);
2. $\mathcal{L}$ has one-step interpolation;
3. $\mathcal{L}$ has uniform interpolation.

Here a pullback is called surjective if it consists of surjective maps. If the modalities of $\mathcal{L}$ are separating and monotone, then preservation of finite surjective weak pullbacks is in fact necessary for one-step interpolation.

As applications of this result, we obtain that neighbourhood logic (i.e. classical modal logic [4]), monotone modal (neighbourhood) logic [4], the relational modal logics K and KD, coalition logic [29], and logics of monoid-weighted transition systems for finite refinable commutative monoids (in particular for finite Abelian groups, even though the latter fail to admit monotone modalities) have uniform interpolation; for neighbourhood logic, coalition logic, and monoid-weighted logics, these results appear to be new.

Related Work

Craig interpolation for monotone modal logic was first proved by Hansen and Kupke [14] and later improved to uniform interpolation by Santocanale and Venema [32]. Craig interpolation (but not uniform interpolation) for coalition logic was proved by Schröder and Pattinson using coalgebraic cutfree sequent systems [28]. Hansen, Kupke, and Pacuit [15] have proved Craig interpolation (but not uniform interpolation) for neighbourhood logic, using semantic methods. Uniform interpolation for coalgebraic modal logic with a generalized Moss modality based on a *quasifunctorial lax lifting* has been shown, for functors preserving finite sets, by Marti in his MSc thesis [20] (and in fact this result has been extended to coalgebraic modal fixpoint logics [21]). Logics based on diagonal-preserving lax liftings (even without assuming quasi-functoriality) satisfy an obvious variant of separation and thus support a generalized Hennessy-Milner theorem, and moreover can be translated into the language of monotone predicate liftings [20]. We leave it as an open problem to determine the relationship between quasifunctoriality and preservation of surjective weak pullbacks in presence of a separating set of monotone predicate liftings. We emphasize that our criteria for interpolation apply also to logics that fail to be separating or admit monotone modalities, hence cannot be phrased in terms of quasifunctorial lax liftings, notable examples of this type being coalition logic, neighbourhood logic, and logics of finite-Abelian-group-weighted transition systems.

In [27], the (coalgebraic) logic of exact covers was introduced; besides a generic Hennessy-Milner theorem and results on completeness and small models, a generic uniform interpolation

theorem was claimed which implies that every rank-1 modal logic with finitely many (not necessarily monotone) modalities has uniform interpolation. We show by means of a counter-example that the latter claim is incorrect; our results help delineate in which cases it can be salvaged.

2 Preliminaries

We assume basic familiarity with category theory (see [1] for an introduction). Throughout, we work over the category Set of sets and functions as the base category. Given a functor $\mathsf{F} : \mathsf{Set} \to \mathsf{Set}$, an F-*coalgebra* is a pair $\mathbb{X} = (X, \xi)$ consisting of a set X (of states) and a function $\xi : X \to \mathsf{F}X$. In the spirit of coalgebraic logic, we use such coalgebras as generic models of modal logics; e.g. Kripke frames can be seen as a coalgebras $\xi : X \to \mathcal{P}X$ for the powerset functor $\mathcal{P}$, as they assign to each state $x \in X$ a set $\xi(x) \in \mathcal{P}(X)$ of successor states. We will later see non-relational examples. We denote by $\mathcal{Q}$ the contravariant powerset functor, which acts on sets by taking powersets and on maps by taking preimage maps ($\mathcal{Q}f(A) = f^{-1}[A]$).

2.1 Set Functors and Weak Pullbacks

The pullback of a cospan $(f, g) = X \xrightarrow{f} Z \xleftarrow{g} Y$ in Set is described as

$$(\mathsf{pb}(f,g), \pi_1, \pi_2) \text{ where } \mathsf{pb}(f,g) = \{(x,y) \in X \times Y \mid f(x) = g(y)\}$$

and π_1, π_2 are the projections to the components.

An important property of set functors in the analysis of coalgebras is weak pullback preservation [31]. A set functor F *preserves weak pullbacks*, or *weakly preserves pullbacks* if it maps pullbacks to *weak pullbacks* (equivalently maps weak pullbacks to weak pullbacks), where a weak pullback is defined categorically like a pullback but requiring only existence, not uniqueness, of mediating morphisms. In an element-wise formulation, $X \xleftarrow{\pi_1} P \xrightarrow{\pi_2} Y$ is a weak pullback of $X \xrightarrow{f} Z \xleftarrow{g} Y$ if whenever $f(x) = g(y)$ for $x \in X$, $y \in Y$, then there exists $p \in P$ (not necessarily unique) such that $\pi_1(p) = x$, $\pi_2(p) = y$. It well-known and easy to see that the identity functor, all constant functors, and the powerset functor preserve weak pullbacks and that the class of weak-pullback preserving functors is closed under products, coproducts, exponentiation with constants, functor composition, and taking finitary parts [31, 37]. In particular, all generalized Kripke polynomial functors (built from powerset, finite powerset, constant functors, and identity by taking products, coproducts, and exponentiation with constants) preserve weak pullbacks. Some negative examples are as follows.

▶ **Example 1.**

1. The *Neighbourhood functor* or double contravariant powerset functor $\mathcal{N} = \mathcal{Q}\mathcal{Q}$ maps a set X to $\mathcal{N}X = \mathcal{Q}\mathcal{Q}X$ and a function $f : X \to Y$ to $\mathcal{N}f(\alpha) = \{A \subseteq Y \mid f^{-1}[A] \in \alpha\}$. This functor does not preserve weak pullbacks [31].
2. The *monotone neighbourhood functor* $\mathcal{M}$ is a subfunctor of the neighbourhood functor $\mathcal{N}$. Given an element $\alpha \in \mathcal{Q}\mathcal{Q}X$, we put

   $$\mathsf{Up}(\alpha) := \{Y \subseteq X \mid Y \supseteq Z \text{ for some } Z \in \alpha\},$$

   and then say that α is *upwards closed* if $\alpha = \mathsf{Up}(\alpha)$. The functor $\mathcal{M}$ is then given on sets X by $\mathcal{M}X = \{\alpha \in \mathcal{Q}\mathcal{Q}(X) \mid \alpha \text{ upwards closed}\}$. Like $\mathcal{N}$, $\mathcal{M}$ does not preserve weak pullbacks [14].
3. Another functor that does not preserve weak pullbacks is F_2^3, defined as a subfunctor of the cubing functor $X \mapsto X^3$ by $\mathsf{F}_2^3X = \{(x_1, x_2, x_3) \in X^3 \mid |\{x_1, x_2, x_3\}| \leq 2\}$ [12].

2.2 Coalgebraic Modal Logic

We briefly recall the syntax and semantics of coalgebraic modal logic.

We fix a countable set V of *propositional variables*. The syntax of a coalgebraic modal logic $\mathcal{L}(\Lambda)$ is then determined by the choice a *modal signature* Λ consisting of modal operators with assigned arities: the set $\mathcal{F}(\Lambda)$ of *(modal)* Λ*-formulas* is defined by the grammar

$$\mathcal{L}(\Lambda) \ni \phi, \psi ::= v \mid \bot \mid \neg\phi \mid \phi \wedge \psi \mid \heartsuit(\phi_1, \cdots, \phi_n),$$

where $v \in V$ and $\heartsuit \in \Lambda$ is an n-ary modality (we deviate slightly from usual practice in coalgebraic modal logic by including propositional variables in the syntax rather then regarding them as nullary modalities; this is in order to facilitate the definition of interpolation). Other Boolean operators $(\top, \vee, \rightarrow, \leftrightarrow)$ are defined in the standard way. We write $\mathsf{rk}(\phi)$ for the *rank* of ϕ, i.e. the maximal nesting depth of modal operators in ϕ.

As indicated above, the type of systems underlying the semantics of $\mathcal{L}(\Lambda)$ is determined by the choice of a set functor F whose coalgebras play the role of frames. The interpretation of the modal operators is then defined in terms of predicate liftings for F:

▶ **Definition 2** (Predicate liftings)**.** An *n-ary predicate lifting* for F is a natural transformation $\lambda : \mathcal{Q}(-)^n \to \mathcal{Q} \circ \mathsf{F}$, where $\mathcal{Q}(-)^n$ denotes the n-fold product of $\mathcal{Q}$ with itself. We say that λ is *monotone* if $\lambda_X(Y_1, ..., Y_n) \subseteq \lambda_X(Z_1, ..., Z_n)$, whenever $Y_i \subseteq Z_i \subseteq X$ for each i. Equivalently, we can describe λ by its *transposite* $\lambda^\flat : \mathsf{F} \to \mathcal{Q}\mathcal{Q}^n$ given by $\lambda^\flat_X(t) = \{(Y_1, ..., Y_n) \in \mathcal{Q}\mathcal{Q}^n X \mid t \in \lambda_X(Y_1, ..., Y_n)\}$.

By the Yoneda lemma, we have the following equivalent description of predicate liftings [34].

▶ **Fact 3.** *The n-ary predicate liftings for F are in one-to-one correspondence with subsets of $\mathsf{F}(2^n)$, where $2 = \{\top, \bot\}$; e.g. for $n = 1$, such a subset U determines a predicate lifting λ by $\lambda_X(A) = \{t \in \mathsf{F}X \mid T\chi_A(t) \in U\}$ where $\chi_A : X \to 2$ is the characteristic map of $A \subseteq X$.*

We then complete the semantic parametrization of $\mathcal{L}(\Lambda)$ by assigning to each n-ary modal operator $\heartsuit \in \Lambda$ an n-ary predicate lifting $[\![\heartsuit]\!]$ for F.

▶ **Definition 4.** An F-*model* (X, ξ, τ) consists of an F-coalgebra $\mathbb{X} = (X, \xi)$ and a valuation $\tau : V \to \mathcal{P}(X)$ of the propositional variables. We then inductively define a satisfaction relation $\vDash$ between states of the model (X, ξ, τ) and formulas of $\mathcal{L}(\Lambda)$ by $x \Vdash v$ iff $x \in \tau(v)$, standard clauses for Boolean connectives, and

$$x \vDash \heartsuit(\phi_1, ..., \phi_n) \text{ iff } \xi(x) \in \heartsuit_X([\![\phi_1]\!], ..., [\![\phi_n]\!]),$$

where $[\![\phi_i]\!] = \{t \mid t \vDash \phi_i\}$. As usual, we say that a formula ϕ is *satisfiable* if there exists a state x in some model such that $x \vDash \phi$, and *valid* if $x \vDash \phi$ for every state x in every F-model.

For readability, we mostly restrict the technical exposition to unary modalities from now on; the extension to finitary modalities is just a matter of adding indices.

▶ **Example 5.**

1. The modal logic K is captured coalgebraically by taking the powerset functor $\mathcal{P}$ as the type functor, $\Lambda = \{\Diamond\}$, and

$$[\![\Diamond]\!]_X(Y) = \{A \in \mathcal{P}X \mid A \cap Y \neq \varnothing\}.$$

2. Neighbourhood logic (or *classical modal logic* [4]) has $\Lambda = \{\Box\}$, interpreted over the neighbourhood functor $\mathcal{N}$ (Example 1.1) by

$$[\![\Box]\!]_X(Y) \coloneqq \{\alpha \in \mathcal{N}X \mid Y \in \alpha\}.$$

Monotone modal (neighbourhood) logic is captured in the same way, replacing $\mathcal{N}$ with the monotone neighbourhood functor $\mathcal{M}$ (Example 1.2).

We fix the data F, Λ, $[\![\heartsuit]\!]$ *of the logic* $\mathcal{L}(\Lambda)$ *from now on.*

The One-Step Logic

Given any set Z, we denote by $\mathsf{Prop}(Z)$ the set of propositional formulas over Z:

$$\mathsf{Prop}(Z) \ni \phi, \psi ::= \bot \mid z \mid \neg\phi \mid \phi \wedge \psi \qquad (z \in Z),$$

and write $\Lambda(Z)$ for the set of formulas $\heartsuit(z_1, \cdots, z_n)$ where $\heartsuit \in \Lambda$ has arity n and $z_1, \cdots, z_n \in Z$. We then define a *one-step formula over* Z to be an element of $\mathsf{Prop}(\Lambda(\mathsf{Prop}(Z)))$. Here, Z will often be a subset of V; also, Z will sometimes be a subset of some powerset $\mathcal{P}X$, in which case we will understand every element of $\mathcal{P}X$ to be interpreted as itself. In general, we interpret both propositional formulas and one-step formulas over Z w.r.t. $\mathcal{P}(X)$-valuations $\tau : Z \to \mathcal{P}(X)$ for some set X: We extend τ to propositional formulas using the Boolean algebra structure of $\mathcal{P}X$, obtaining for $\phi \in \mathsf{Prop}(Z)$ a subset

$$\phi\tau \in \mathcal{P}X.$$

We write $X \vDash \phi\tau$ if $\phi\tau = X$. We then define the extension

$$\psi\tau \in \mathcal{P}(\mathsf{F}X)$$

of a one-step formula $\psi \in \mathsf{Prop}(\Lambda(\mathsf{Prop}(Z)))$ recursively by the evident clauses for Boolean connectives, and

$$(\heartsuit\phi)\tau = [\![\heartsuit]\!]_X(\phi\tau).$$

When $Z \subseteq \mathcal{P}(X)$ and τ is just subset inclusion, we omit τ from the notation, so $\psi \in \mathsf{Prop}(\Lambda(\mathcal{P}X))$ denotes both a one-step formula and its interpretation in $\mathcal{P}(\mathsf{F}X)$; we then write $\mathsf{F}X \vDash \psi$ if the interpretation of ψ is all of $\mathsf{F}X$.

▶ **Definition 6.** A one-step formula $\psi \in \mathsf{Prop}(\Lambda(\mathsf{Prop}(Z)))$ is *(one-step) satisfiable over* $\tau : Z \to \mathcal{P}(X)$ if $\psi\tau \neq \emptyset$, and *(one-step) satisfiable* if ψ is one-step satisfiable over τ for some τ. Dually, ψ is *(one-step) valid (over* τ*)* if $\neg\psi$ is (one-step) unsatisfiable (over τ). We write $\mathsf{F}X, \tau \vDash \psi$ if ψ is one-step valid over τ, and $\vDash \psi$ if ψ is one-step valid.

We will need the following pieces of terminology and notation:

▶ **Definition 7.** For a map $f : X \to Y$, we write σ_f for the substitution mapping $A \in \mathcal{P}(X)$ to $f[A]$ (e.g. in one-step formulas of type $\mathsf{Prop}(\Lambda(\mathcal{P}(X)))$), and $\sigma_{f^{-1}}$ for the substitution mapping $B \in \mathcal{P}(Y)$ to $f^{-1}[B]$. A set $A \in \mathcal{P}(X)$ is *f-invariant* if $f^{-1}[f[A]] = A$.

Clearly, all sets of the form $f^{-1}[B]$ are f-invariant, i.e. the f-invariant sets are precisely those of the form $f^{-1}[B]$. The f-invariant sets form a Boolean subalgebra of $\mathcal{P}(X)$. (In fact, for finite X, Boolean subalgebras of $\mathcal{P}(X)$ are in bijection with equivalence relations on X, so *every* Boolean subalgebra of $\mathcal{P}(X)$ consists of the f-invariant sets for a suitable f.)

▶ **Definition 8.** We denote by $\mathcal{S}(\mathfrak{A})$ the set of *atoms* of a finite Boolean algebra $\mathfrak{A}$, i.e. its minimal non-bottom elements, and $\mathsf{can}_{\mathfrak{A}}$ for the canonical isomorphism $\mathfrak{A} \to \mathcal{P}(\mathcal{S}(\mathfrak{A}))$. Given a subalgebra $\mathfrak{A}_0$ of $\mathfrak{A}$, we have a *canonical projection* $\mathcal{S}(\mathfrak{A}) \to \mathcal{S}(\mathfrak{A}_0)$.

The following lemmas are straightforward consequences of naturality of predicate liftings:

▶ **Lemma 9.** *Given a finite Boolean subalgebra $\mathfrak{A}$ of $\mathcal{P}X$ for a set X, $\phi \in \mathsf{Prop}(\Lambda(\mathfrak{A}))$ is satisfiable iff $\phi\,\mathsf{can}_{\mathfrak{A}}$ is satisfiable. Dually, ϕ is valid ($\mathsf{F}X \vDash \phi$) iff $\phi\,\mathsf{can}_{\mathfrak{A}}$ is valid ($\mathsf{F}\mathcal{S}(\mathfrak{A}) \vDash \phi\,\mathsf{can}_{\mathfrak{A}}$).*

▶ **Lemma 10.** *Let $\mathfrak{A}_0 \subseteq \mathfrak{A}_1$ be finite Boolean subalgebras of $\mathcal{P}X$ for a set X, let $f : \mathcal{S}(\mathfrak{A}_1) \to \mathcal{S}(\mathfrak{A}_0)$ be the canonical projection, let $\phi \in \mathsf{Prop}(\Lambda(\mathfrak{A}_0))$, and let $t \in \mathsf{F}(\mathcal{S}(\mathfrak{A}_1))$. Then $t \in \phi\,\mathsf{can}_{\mathfrak{A}_1}$ iff $\mathsf{F}f(t) \in \phi\,\mathsf{can}_{\mathfrak{A}_0}$.*

Separation and Maximally Satisfiable Sets

The key condition ensuring that $\mathcal{L}(\Lambda)$ satisfies the Hennessy-Milner property, i.e. distinguishes non-bisimilar states, is *separation* [26, 34]:

▶ **Definition 11** (Separation)**.** We say that Λ is *separating* if for each set X, the family of maps $(\llbracket\heartsuit\rrbracket^\flat_X : \mathsf{F}X \to \mathcal{Q}\mathcal{Q}X = \mathcal{N}X)_{\heartsuit\in\Lambda}$ is jointly injective.

We proceed to define the *MSS-functor* (for *maximally one-step satisfiable sets*) from F and Λ. (A related functor using *maximally one-step consistent* sets has been used to show that every rank-1 modal logic has a coalgebraic semantics [36].)

▶ **Definition 12.** A set $\Phi \subseteq \mathsf{Prop}(\Lambda(\mathcal{P}(X)))$ is *one-step satisfiable* if the intersection of the interpretations of the formulas in Φ is non-empty, and *maximally one-step satisfiable* if Φ is maximal among such sets. The *MSS-functor* M^Λ_{F} is given by $M^\Lambda_{\mathsf{F}}X$ being the set of maximally one-step satisfiable subsets of $\mathsf{Prop}(\Lambda(\mathcal{P}X))$, and $M^\Lambda_{\mathsf{F}}f(\Phi) = \{\phi \in \mathsf{Prop}(\Lambda(\mathcal{P}(Y))) \mid \phi\sigma_{f^{-1}} \in \Phi\}$ for $f : X \to Y$.

The following lemma allows us to identify F with its MSS-functor whenever Λ is separating.

▶ **Lemma 13.** *If Λ is separating, then F and M^Λ_{F} are isomorphic.*

3 Surjective Weak Pullbacks

We proceed to introduce the key semantic interpolation criterion, preservation of *surjective* weak pullbacks. We record explicitly:

▶ **Definition 14.** A pullback of a cospan (f, g) of maps (in Set) is *surjective* if both f and g are surjective, and *finite* if all involved sets are finite. A functor *preserves (finite) surjective weak pullbacks* if it maps (finite) surjective pullbacks to weak pullbacks.

Recall that under the axiom of choice, every set functor preserves surjective maps. Also, surjective maps are stable under pullbacks, so all morphisms in a surjective pullback are surjective. Non-empty binary Cartesian products $X \times Y$ are surjective pullbacks of $X \to 1 \leftarrow Y$. Moreover, the kernel pair of a map $f : X \to Y$ is a surjective pullback of the codomain restriction $X \to f[X]$.

For finitary functors, the finiteness restriction in the preservation condition is immaterial:

▶ **Lemma 15.** *If F is finitary, then F preserves (surjective) weak pullbacks iff F preserves finite (surjective) weak pullbacks.*

Of course, every functor that preserves weak pullbacks also preserves surjective weak pullbacks, e.g. the (finite or unrestricted) powerset functor, and more generally all Kripke polynomial functors. Two negative examples are as follows.

▶ **Example 16.**

1. The neighbourhood functor $\mathcal{N}$ fails to preserve finite surjective weak pullbacks. To see this, consider the pullback of the following functions as in [31]. Let $X = \{a_1, a_2, a_3\}$, $Y = \{b_1, b_2, b_3\}$ and $Z = \{c_1, c_2\}$ and define surjective maps $f : X \to Z$ and $g : Y \to Z$ as follows: $f(a_1) = f(a_2) = c_1$, $f(a_3) = c_2$, $g(b_1) = c_1$ and $g(b_2) = g(b_3) = c_2$.
2. The functor F_2^3 fails to preserve finite surjective weak pullbacks. For a counterexample consider a surjective cospan (f, g) with $f = g$ being the constant map $\{a, b\} \to \{b\}$. For $u = (b, b, a)$ and $v = (a, b, b)$, it is impossible to find a $w \in \mathsf{F}_2^3\mathsf{pb}(f, g)$ such that $\mathsf{F}_2^3\pi_1(w) = u$ and $\mathsf{F}_2^3\pi_2(w) = v$.

We proceed to see examples that fail to preserve weak pullbacks but do preserve surjective weak pullbacks.

The Monotone Neighbourhood Functor

The monotone neighbourhood functor $\mathcal{M}$ does not preserve all weak pullbacks (Example 1.2). However:

▶ **Proposition 17.** *The monotone neighbourhood functor $\mathcal{M}$ preserves surjective weak pullbacks.*

The proof is facilitated by the following fact:

▶ **Lemma and Definition 18** (Compatibility). *Let*

$$\begin{array}{ccc} P & \xrightarrow{\pi_1} & X \\ {\scriptstyle \pi_2}\downarrow & & \downarrow{\scriptstyle f} \\ Y & \xrightarrow[g]{} & Z \end{array} \tag{1}$$

be a surjective pullback, and let $\alpha_1 \in \mathcal{M}X$, $\alpha_2 \in \mathcal{M}Y$. Then $\mathcal{M}f(\alpha_1) = \mathcal{M}g(\alpha_n)$ iff α_1 and α_2 are compatible*, i.e. for every $U \in \alpha_1$ we have $\pi_2[\pi_1^{-1}[U]] \in \alpha_2$ and symmetrically.*

Proof (Proposition 17, Sketch). Given a surjective pullback (1) and compatible $\alpha_1 \in \mathcal{M}X$, $\alpha_2 \in \mathcal{M}Y$, it is straightforward to show that

$$\beta = \mathsf{Up}(\{\pi_1^{-1}[U] \mid U \in \alpha_1\} \cup \{\pi_2^{-1}[V] \mid V \in \alpha_2\}) \in \mathcal{M}P$$

satisfies $\mathcal{M}\pi_1(\beta) = \alpha_1$ and $\mathcal{M}\pi_2(\beta) = \alpha_2$. ◀

Monoid-weighted Functors

Given a commutative monoid M (which we write additively), the *monoid-weighted functor* S_M is defined by taking $\mathsf{S}_M X$ to be the set of finitely supported functions $X \to M$ (i.e. functions that vanish almost everywhere), and $\mathsf{S}_M f(\mu) = \lambda y.\ \sum_{f(x)=y} \mu(x)$ for $f : X \to Y$ and $\mu \in \mathsf{S}_M X$. Examples of monoid-weighted functors include the free Abelian groups functor ($M = \mathbb{Z}$), the free vector space functor ($M = \mathbb{R}$), the finite multiset functor ($M = \mathbb{N}$), and the finite powerset functor ($M = 2 = \{\bot, \top\}$ with $+$ being disjunction).

▶ **Definition 19** (Refinability). [13] A commutative monoid M is *refinable* if whenever $\sum_{i=1}^{n} a_i = \sum_{j=1}^{k} b_j$ for $a_1, \dots, a_n, b_1, \dots, b_k \in M$, $n, k \geq 1$, then there exists an $n \times k$-matrix over M with row sums a_i and column sums b_j.

As shown by Gumm and Schröder [13], S_M preserves weak kernel pairs iff M is refinable. In fact, refinability already ensures preservation of all weak surjective pullbacks:

▶ **Lemma 20.** *The functor S_M preserves weak surjective pullbacks iff M is refinable.*

Given that a) weak pullback preserving finitary functors are known to admit separating sets of monotone predicate liftings [18], and b) the monotone neighbourhood functor itself preserves surjective weak pullbacks but not all weak pullbacks, it is tempting to conjecture that preservation of surjective weak pullbacks is already sufficient for existence of a separating set of monotone predicate liftings. This is not true, however:

▶ **Definition 21.** A commutative monoid M is *positive* if for $a, b \in M$, $a + b = 0$ implies $a = b = 0$.

▶ **Proposition 22.** *Let M be refinable. Then S_M has a separating set of monotone predicate liftings iff M is positive.*

That is, every commutative monoid that is refinable but not positive gives rise to a monoid-weighted functor that preserves surjective weak pullbacks but does not admit a separating set of monotone predicate liftings. One class of such commutative monoids are the non-trivial Abelian groups: they clearly fail to be positive, and are easily seen to be refinable [13].

4 One-Step Interpolation

We proceed to develop our notion of one-step interpolation, and its relationship to preservation of surjective weak pullbacks.

▶ **Assumption 23.** From here on, we assume throughout that *the modal signature Λ is finite.*

In a nutshell, $\mathcal{L}(\Lambda)$ has one-step interpolation if adding one layer of modalities preserves interpolation:

▶ **Definition 24.** Two Boolean subalgebras $\mathfrak{A}_1$, $\mathfrak{A}_2$ of $\mathcal{P}(X)$ for a set X are *interpolable* if whenever $A \subseteq B$ for $A \in \mathfrak{A}_1$ and $B \in \mathfrak{A}_2$, then there exists $C \in \mathfrak{A}_1 \cap \mathfrak{A}_2$ such that $A \subseteq C$ and $C \subseteq B$. We say that $\mathcal{L}(\Lambda)$ has *one-step interpolation* if given interpolable $\mathfrak{A}_1$, $\mathfrak{A}_2$ and $\phi \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1))$, $\psi \in \mathsf{Prop}(\Lambda(\mathfrak{A}_2))$ such that $\mathsf{F}X \vDash \phi \to \psi$, there is always an *interpolant* $\rho \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1 \cap \mathfrak{A}_2))$ such that $\mathsf{F}X \vDash \phi \to \rho$ and $\mathsf{F}X \vDash \rho \to \psi$. Moreover, $\mathcal{L}(\Lambda)$ has *uniform one-step interpolation* if the interpolant can be made to depend only on ϕ and $\mathfrak{A}_1 \cap \mathfrak{A}_2 =: \mathfrak{A}_0$; it is then called a *uniform $\mathfrak{A}_0$-interpolant* of ϕ.

It is in fact not hard to see that under Assumption 23, these two notions coincide, so we refer to them just as *one-step interpolation*:

▶ **Lemma 25.** *The logic $\mathcal{L}(\Lambda)$ has one-step uniform interpolation iff $\mathcal{L}(\Lambda)$ has one-step interpolation.*

Proof. *'Only if'*: trivial. *'If'*: One-step interpolation implies that, given data as in Definition 24, the formula

$$i(\phi) = \bigwedge \{\rho \in \mathsf{Prop}(\Lambda(\mathsf{Prop}(\mathfrak{A}_0))) \mid \mathsf{F}X \vDash \phi \to \rho\},$$

which is effectively finite because Λ is finite, is a uniform $\mathfrak{A}_0$-interpolant of ϕ. ◀

▶ **Remark 26.** In any logic supporting the requisite propositional connectives, the set of formulas ϕ having a uniform interpolant is easily seen to be closed under disjunction, and similarly the set of pairs of formulas ϕ, ψ such that an interpolant between ϕ and ψ exists is closed under disjunction in ϕ and under conjunction in ψ. When establishing (one-step) uniform interpolation for ϕ, or (one-step) interpolation between ϕ and ψ, we can therefore assume that ϕ is a conjunctive clause over modalized formulas and that ψ is a disjunctive clause over modalized formulas.

Our first positive example is neighbourhood logic:

▶ **Example 27.** Neighbourhood logic has one-step interpolation (and hence, by Lemma 25, uniform one-step interpolation). To see this, let $\mathfrak{A}_1, \mathfrak{A}_2$ be interpolable Boolean subalgebras of $\mathcal{P}(X)$, let ϕ be a conjunctive clause over $\Lambda(\mathfrak{A}_1)$, and let ψ be a disjunctive clause over $\Lambda(\mathfrak{A}_2)$ such that $\mathsf{F}X \vDash \phi \to \psi$ (this case suffices by Remark 26). We can assume w.l.o.g. that $\mathsf{F}X \nvDash \neg\phi$ and $\mathsf{F}X \nvDash \psi$. Then $\mathsf{F}X \vDash \phi \to \psi$ implies that ϕ contains a conjunct $\epsilon \Box A$ and ψ a disjunct $\epsilon \Box B$, with ϵ representing either nothing or negation, such that $A = B$. Then $\epsilon \Box A$ interpolates between ϕ and ψ.

Preservation of surjective weak pullbacks is sufficient for uniform one-step interpolation:

▶ **Lemma 28.** *Let Λ be separating, and let F preserve finite surjective weak pullbacks. Then $\mathcal{L}(\Lambda)$ has one-step uniform interpolation.*

Proof. Let $\mathfrak{A}_0 \subseteq \mathfrak{A}_1$ be finite Boolean subalgebras of $\mathcal{P}X$, and let $\phi \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1))$. We show that

$$i(\phi) = \bigwedge\{\rho \in \mathsf{Prop}(\Lambda(\mathfrak{A}_0)) \mid \mathsf{F}X \vDash \phi \to \rho\}$$

(effectively a finite formula) is a uniform $\mathfrak{A}_0$-interpolant for ϕ. Dually, we show for $\psi \in \mathsf{Prop}(\Lambda(\mathfrak{A}_2))$ with $\mathfrak{A}_1$, $\mathfrak{A}_2$ interpolable, $\mathfrak{A}_1 \cap \mathfrak{A}_2 \subseteq \mathfrak{A}_0$, and $i(\phi) \wedge \psi$ satisfiable that also $\phi \wedge \psi$ is satisfiable. By Lemma 9, we have $s \in \mathsf{F}(\mathcal{S}(\mathfrak{A}_2))$ such that $s \vDash (i(\phi) \wedge \psi)\,\mathsf{can}_{\mathfrak{A}_2}$. Let θ be the $\mathsf{Prop}(\Lambda(\mathfrak{A}_1 \cap \mathfrak{A}_2))$-theory $\theta = \bigwedge\{\rho \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1 \cap \mathfrak{A}_2)) \mid s \vDash \rho\,\mathsf{can}_{\mathfrak{A}_2}\}$ of s. Then $\phi \wedge \theta$ is satisfiable: otherwise, $\mathsf{F}X \vDash \phi \to \neg\theta$, so $\mathsf{F}X \vDash i(\phi) \to \neg\theta$ by definition of $i(\phi)$, which by Lemma 9 contradicts $s \vDash (i(\phi) \wedge \theta)\,\mathsf{can}_{\mathfrak{A}_2}$.

Again by Lemma 9, we thus have $t \in \mathsf{F}(\mathcal{S}(\mathfrak{A}_1))$ such that $t \vDash (\phi \wedge \theta)\,\mathsf{can}_{\mathfrak{A}_1}$. Let $\mathfrak{A}$ be the Boolean subalgebra of $\mathcal{P}X$ generated by $\mathfrak{A}_1 \cup \mathfrak{A}_2$. Then the diagram

$$\begin{array}{ccc} \mathcal{S}(\mathfrak{A}) & \xrightarrow{\pi_1} & \mathcal{S}(\mathfrak{A}_1) \\ {\scriptstyle \pi_2}\downarrow & & \downarrow{\scriptstyle f} \\ \mathcal{S}(\mathfrak{A}_2) & \xrightarrow[g]{} & \mathcal{S}(\mathfrak{A}_1 \cap \mathfrak{A}_2), \end{array}$$

where all maps are canonical projections, is a finite surjective pullback because $\mathfrak{A}_1$, $\mathfrak{A}_2$ are interpolable, hence weakly preserved by F. We claim that $\mathsf{F}f(t) = \mathsf{F}g(s)$: indeed, both sides satisfy $\theta\,\mathsf{can}_{\mathfrak{A}_1 \cap \mathfrak{A}_2}$ by Lemma 10, and since θ is a complete $\mathsf{Prop}(\Lambda(\mathfrak{A}_1 \cap \mathfrak{A}_2))$-theory, equality follows by separation. It follows that we have $u \in \mathsf{F}(\mathcal{S}(\mathfrak{A}))$ such that $\mathsf{F}\pi_1(u) = t$ and $\mathsf{F}\pi_2(u) = s$. Again by Lemma 10, $u \vDash \phi\,\mathsf{can}_{\mathfrak{A}}$ and $u \vDash \psi\,\mathsf{can}_{\mathfrak{A}}$, so by Lemma 9, $\phi \wedge \psi$ is satisfiable. ◀

The example of neighbourhood logic (Examples 27 and 16.1) shows that the converse of Lemma 28 does not hold in general. It does however hold in the monotone case:

▶ **Lemma 29.** *Let $\mathcal{L}(\Lambda)$ be monotone and separating and have one-step interpolation. Then F preserves finite surjective weak pullbacks.*

The proof relies on invariant sets, and uses the following lemma (which can be seen as a rewording of Lemma 9):

▶ **Lemma 30.** *Let $f : X \to Y$ be surjective, let $\mathfrak{A}$ denote the subalgebra of $\mathcal{P}(X)$ consisting of the f-invariant sets, and let $\phi \in \mathsf{Prop}(\Lambda(\mathfrak{A}))$. Then $\mathsf{F}X \vDash \phi$ iff $\mathsf{F}Y \vDash \phi\sigma_f$.*

Proof (Lemma 29, Sketch). Let $X \xleftarrow{\pi_1} P \xrightarrow{\pi_2} Y$ be a finite surjective pullback of $X \xrightarrow{f} Z \xleftarrow{g} Y$ as in Diagram (1). As indicated in Section 2, we can identify F with its MSS functor, i.e. we assume that $\mathsf{F}X$ consists of maximally satisfiable subsets $\Phi \subseteq \mathsf{Prop}(\Lambda(\mathcal{P}(X)))$. In this reading, we are given $\Phi_1 \in \mathsf{F}X$ and $\Phi_2 \in \mathsf{F}Y$ such that $\mathsf{F}f(\Phi_1) = \mathsf{F}g(\Phi_2)$, which by a straightforward generalization of Lemma 18 means that Φ_1 and Φ_2 are *compatible*, i.e.

$$\phi \in \Phi_2 \text{ implies } \phi\sigma_{\pi_2^{-1}}\sigma_{\pi_1} \in \Phi_1$$

and symmetrically. We have to show that there exists $\Phi \in \mathsf{F}R$ such that $F\pi_1(\Phi) = \Phi_1$ and $F\pi_2(\Phi) = \Phi_2$, i.e.

$$\phi \in \Phi_1 \iff \phi\sigma_{\pi_1^{-1}} \in \Phi \tag{2}$$

and correspondingly for Φ_2. In (2), '$\Rightarrow$' is sufficient, because the logic has negation. That is, we have to show that the set $\{\phi\sigma_{\pi_1^{-1}} \mid \phi \in \Phi_1\} \cup \{\phi\sigma_{\pi_2^{-1}} \mid \phi \in \Phi_2\}$ is one-step satisfiable. Since Φ_1 and Φ_2 are effectively finite and closed under conjunctions, it thus suffices to show that whenever $\phi_1 \in \Phi_1$ and $\phi_2 \in \Phi_2$, then

$$\phi = \phi_1\sigma_{\pi_1^{-1}} \wedge \phi_2\sigma_{\pi_2^{-1}}$$

is one-step satisfiable. Assume the contrary; then $\phi_1\sigma_{\pi_1^{-1}} \to \neg\phi_2\sigma_{\pi_2^{-1}}$ is one-step valid. Now let $\mathfrak{A}_1$ denote the Boolean subalgebra of $\mathcal{P}(R)$ consisting of the π_1-invariant sets, correspondingly $\mathfrak{A}_2$ for the π_2-invariant sets. One checks that $\mathfrak{A}_1$, $\mathfrak{A}_2$ are interpolable. Since $\mathcal{L}(\Lambda)$ has one-step interpolation, we therefore find $\rho \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1 \cap \mathfrak{A}_2))$ such that $R \vDash \phi_1\sigma_{\pi_1^{-1}} \to \rho$ and $R \vDash \rho \to \neg\phi_2\sigma_{\pi_2^{-1}}$. Using surjectivity of the π_i, Lemma 30, and compatibility, we can derive $\rho\sigma_{\pi_1} \in \Phi_1$, $\rho\sigma_{\pi_2} \in \Phi_2$, and eventually $\neg\phi_2 \in \Phi_2$, contradicting satisfiability of Φ_2. ◀

5 Uniform Interpolation

We now relate one-step interpolation to interpolation for the full logic. Recall from Section 2.2 that we work in a language with propositional variables. Given a set $V_0 \subseteq V$ of propositional variables, we write $\mathcal{F}(\Lambda, V_0)$ for the set of Λ-formulas mentioning only propositional atoms from V_0, and put

$$\mathcal{F}_n(\Lambda, V_0) = \{\phi \in \mathcal{F}(\Lambda, V_0) \mid \mathsf{rk}(\phi) \leq n\}.$$

For a state x in some model, we put

$$\mathsf{Th}^n_{V_0}(x) = \{\rho \in \mathcal{F}_n(\Lambda, V_0) \mid x \vDash \rho\}$$

(eliding the model, which will always be clear from the context). Since Λ is assumed to be finite, we have

▶ **Lemma 31.** *For finite V_0, $\mathcal{F}_n(\Lambda, V_0)$ is finite up to logical equivalence.*

We record explicitly:

▶ **Definition 32.** We say that $\mathcal{L}(\Lambda)$ has *interpolation* if whenever $\vDash \phi \to \psi$ for $\phi \in \mathcal{F}(\Lambda, V_1)$ and $\psi \in \mathcal{F}(\Lambda, V_2)$, then there exists an *interpolant* $\rho \in \mathcal{F}(\Lambda, V_1 \cap V_2)$ such that $\vDash \phi \to \rho$ and $\vDash \rho \to \psi$; and $\mathcal{L}(\Lambda)$ has *uniform interpolation* if the interpolant ρ can be made to depend only on $V_0 := V_1 \cap V_2$. We then call ρ a *uniform* V_0*-interpolant* of ϕ.

We do not currently know whether one-step interpolation in the strong sense of Definition 24 is necessary for $\mathcal{L}(\Lambda)$ to have interpolation. However, a weaker version of one-step interpolation is necessary:

▶ **Lemma 33.** *If* $\mathcal{L}(\Lambda)$ *has interpolation, then the one-step logic* $\mathsf{Prop}(\Lambda(\mathsf{Prop}(V)))$ *has interpolation.*

This can be used to disprove interpolation in some examples (contradicting [27] as indicated in the introduction):

▶ **Example 34.** Let $\mathcal{N}_\vee$ be the subfunctor of the neighbourhood functor $\mathcal{N}$ defined by

$$\mathcal{N}_\vee X = \{\alpha \in \mathcal{N}X \mid \forall A, B \subseteq X.\, A \cup B = X \Rightarrow (A \in \alpha \vee B \in \alpha)\},$$

and interpret the modality $\Box$ over $\mathcal{N}_\vee$ like over $\mathcal{N}$. Take $V_1 = \{p, q\}$, $V_2 = \{r, p\}$. Then the implication $\neg \Box (p \vee q) \to \Box(\neg p \vee r)$ is valid but has no interpolant in $\mathsf{Prop}(\{\Box\}(\mathsf{Prop}\{p\}))$.

As to sufficiency, we have

▶ **Theorem 35.** *If* $\mathcal{L}(\Lambda)$ *has one-step interpolation then* $\mathcal{L}(\Lambda)$ *has uniform interpolation.*

Proof (Sketch). Induction on the rank, proving the stronger claim that the rank of the uniform interpolant of ϕ is at most $\mathsf{rk}(\phi)$. Let $\phi \in \mathcal{L}_n(\Lambda, V_1)$, and let $V_0 \subseteq V_1$. We claim that

$$i(\phi) = \bigwedge\{\phi' \in \mathcal{F}_n(\Lambda, V_0) \mid \vDash \phi \to \phi'\}$$

(by Lemma 31, effectively a finite formula) is a uniform V_0-interpolant for ϕ. The proof reduces straightforwardly to showing that, given $\psi \in \mathcal{F}(\Lambda, V_2)$ where $V_1 \cap V_2 \subseteq V_0$ and models $D = (Y, \zeta, \tau_2)$, $C = (X, \xi, \tau_1)$ and $y_0 \in Y$, $x_0 \in X$ such that $y_0 \vDash_D i(\phi) \wedge \psi$ and $x_0 \vDash_C \phi \wedge \mathsf{Th}_{V_0}(y_0)$, the formula $\phi \wedge \psi$ is satisfiable.

Using a minor variation of standard model constructions in coalgebraic modal logic [33, 35, 24], we can assume that C, D are finite dags in which all states have a well-defined *height* (distance from any initial state in a supporting Kripke frame), with x_0 and y_0 being initial states whose depth (length of the longest path starting at x_0 and y_0, respectively) equals the rank of the relevant formulas, and in which every state x of height $n-k$ in C is uniquely determined (among the states of height $n-k$) by $\mathsf{Th}^k_{V_1}(x)$, correspondingly for $y \in D$ and $\mathsf{Th}^k_{V_2}(x)$. Moreover, we can assume that the models are *canonical*, i.e. every maximally satisfiable subset of $\mathcal{F}_k(\Lambda, V_1)$ is indeed satisfied at a unique state of height $n-k$ of C, as by Lemma 31, there are only finitely many such sets; correspondingly for D and $\mathcal{F}_k(\Lambda, V_2)$.

We now construct a model $E = (Z, \gamma, \tau)$ of $\phi \wedge \psi$ as follows. We put

$$Z = \{(x, y) \in X \times Y \mid n \geq \mathsf{ht}(x) = \mathsf{ht}(y) =: k, \mathsf{Th}^{n-k}_{V_0}(x) = \mathsf{Th}^{n-k}_{V_0}(y)\}$$
$$\cup \{y \in Y \mid \mathsf{ht}(y) > n\},$$

denoting the first part by Z_0 and the second by Z_1, and their height-k levels by Z^k_0, Z^k_1, Z^k, respectively. Note that $(x_0, y_0) \in Z_0$. It is straightforward to define the valuation τ on Z. Moreover, we define a coalgebra structure $\gamma : Z \to \mathsf{F}Z$ such that $\gamma(z) \in \mathsf{F}Z^{k+1}$ for $z \in Z^k$. We put $\gamma(y) = \zeta(y) \in \mathsf{F}Z_1 \subseteq \mathsf{F}Z$ for $y \in Z_1$ (using that w.l.o.g. F preserves inclusions [2]), and on

states $(x,y) \in Z_0$ of maximal height n by $\gamma(x,y) = \zeta(y)$. On the rest of Z_0 we define γ by a *coherence* requirement: By construction of Z, we have a well-defined *pseudo-satisfaction* relation $\models^0$ on Z given by

$$(x,y) \models^0 \rho \iff \begin{cases} x \models_C \rho & (\rho \in \mathcal{F}_{n-\mathsf{ht}(x)}(\Lambda, V_1)) \\ y \models_D \rho & (\rho \in \mathcal{F}_{\mathsf{rk}(\psi)-\mathsf{ht}(y)}(\Lambda, V_2)) \end{cases}$$
$$y \models^0 \rho \iff y \models_D \rho \qquad (\rho \in \mathcal{F}_{\mathsf{rk}(\psi)-\mathsf{ht}(y)}(\Lambda, V_2))$$

For a $\rho \in \mathcal{F}_n(\Lambda, V_1) \cup \mathcal{F}_{\mathsf{rk}(\psi)}(\Lambda, V_2)$, we then have the *pseudo-extension* $\hat{\rho} \subseteq Z$ defined by

$$\hat{\rho} = \{z \in Z \mid \mathsf{rk}(\rho) \leq n - \mathsf{ht}(z), z \models^0 \rho\}.$$

Then we say that γ is *coherent* if for $\heartsuit\rho \in \mathcal{F}_{n-k}(\Lambda, V_1) \cup \mathcal{F}_{n-k}(\Lambda, V_2)$, $k \leq n$, and $(x,y) \in Z_0^k$,

$$\gamma(x,y) \models \heartsuit(\hat{\rho} \cap Z_0^{k+1}) \iff (x,y) \models^0 \heartsuit\rho. \tag{3}$$

For $i = 0,1,2$, let $\mathfrak{A}_i$ be the Boolean subalgebra of $\mathcal{P}Z_0^{k+1}$ consisting of the sets of the form $\hat{\rho} \cap Z_0^{k+1}$ for $\rho \in \mathcal{F}_{n-k-1}(\Lambda, V_i)$. Then, of course, $\mathfrak{A}_1 \cap \mathfrak{A}_2 \subseteq \mathfrak{A}_0$, and by induction, $\mathfrak{A}_1$, $\mathfrak{A}_2$ are interpolable. We define $\phi_0 \in \mathsf{Prop}(\Lambda(\mathfrak{A}_1))$ and $\psi_0 \in \mathsf{Prop}(\Lambda(\mathfrak{A}_2))$ by

$$\phi_0 = \bigwedge\{\epsilon\heartsuit(\hat{\rho} \cap Z_0^{k+1}) \mid (x,y) \models^0 \epsilon\heartsuit\rho, \rho \in \mathcal{F}_{n-k-1}(\Lambda, V_1), \epsilon \in \{\cdot, \neg\}\}$$
$$\psi_0 = \bigwedge\{\epsilon\heartsuit(\hat{\rho} \cap Z_0^{k+1}) \mid (x,y) \models^0 \epsilon\heartsuit\rho, \rho \in \mathcal{F}_{n-k-1}(\Lambda, V_2), \epsilon \in \{\cdot, \neg\}\}.$$

By Lemma 25, ϕ_0 has a uniform $\mathfrak{A}_0$-interpolant $i(\phi_0)$, and by showing satisfiability of $i(\phi_0) \wedge \psi_0$, one establishes that $\phi_0 \wedge \psi_0$ is satisfiable, which means that $\gamma(x,y)$ satisfying (3) exists. Then, we have by induction on $\rho \in \mathcal{F}_n(\Lambda, V_1) \cup \mathcal{F}_{\mathsf{rk}(\psi)}(\Lambda, V_2)$ that $z \models_E \rho$ iff $z \models^0 \rho$ for $\mathsf{ht}(z) = k$ and $\mathsf{rk}(\rho) \leq n - k$; so in particular $z_0 = (x_0, y_0) \models \phi$ and $z_0 \models \psi$, as required. ◀

▶ **Remark 36.** Canonical models in the sense of the above proof sketch in fact bear a strong resemblance to models based on the stages of the final sequence of functors of the type $\mathcal{P}(V_i) \times \mathsf{F}$, $i = 0,1$ (e.g. [25]). This indicates in particular that the proof may eventually be made to bear a relationship, via duality, with Ghilardi's method of graded modal algebras [10].

As indicated in the introduction, our results can be summed up as follows:

▶ **Theorem 37.** *Let Λ be finite. Then the following properties imply each other in sequence:*

1. *Λ is separating and the type functor F preserves finite surjective weak pullbacks.*
2. *$\mathcal{L}(\Lambda)$ has one-step interpolation*
3. *$\mathcal{L}(\Lambda)$ has uniform interpolation*
4. *$\mathcal{L}(\Lambda)$ has interpolation*
5. *The one-step logic $\mathsf{Prop}(\Lambda(\mathsf{Prop}(V)))$ has interpolation.*

Moreover, if Λ is monotone and separating, then 2. implies 1.

We note that if Λ is finite and separating, then F preserves finite sets. As indicated above, we suspect but cannot currently prove that 5 implies 2, which would make items 2–5 equivalent. From Theorem 37, we obtain uniform interpolation for the following concrete logics:

▶ **Example 38.**

1. Whenever F preserves weak pullbacks and Λ is finite and separating, then $\mathcal{L}(\Lambda)$ has uniform interpolation. This case is covered already in [20], see Remark 39. In particular, we obtain that the modal logics K and KD have uniform interpolation, thus reproving previous results [10, 38].

2. Since the monotone neighbourhood functor preserves surjective weak pullbacks (Section 3), we obtain that monotone modal logic has uniform interpolation, again reproving a previous result [32].
3. If M is a finite refinable monoid, then the monoid-weighted functor S_M (Section 3) preserves surjective weak pullbacks, so that any rank-1 modal logic that is expressive (i.e. separating) for S_M has uniform interpolation, such as the logic with modalities $[m]$ for $m \in M$, interpreted by the predicate lifting given by $[m]_X(A) = \{\mu \in \mathsf{S}_M \mid \sum_{x\in A} \mu(x) = m\}$. This holds in particular when M is a finite Abelian group, in which case S_M does not have a separating set of monotone predicate liftings so that this case is not covered by existing generic results [20]. If we take $M = \mathbb{Z}/n\mathbb{Z}$, then the modalities $[m]$ described above are modulo-constraints as found in Presburger modal logic [7]: $[m]\phi$ says that the number of successors of the current state (counting multiplicities) equals m modulo n.
4. Neighbourhood logic fails to preserve surjective weak pullbacks (Example 16) but does have one-step interpolation (Example 27), so we obtain that neighbourhood logic has uniform interpolation.
5. One-step interpolation has been proved, in slightly different terms, for coalition logic [28], so that our results improve the known interpolation result for coalition logic [28] to uniform interpolation.

▶ **Remark 39.** We conclude with a more detailed discussion of the relationship between our results and results on the logic of quasi-functorial lax liftings. Glossing over the ramifications of the axiomatics, a *diagonal-preserving lax lifting* L for a set functor T [22] extends T to act also on relations, satisfying monotonicity w.r.t. inclusion of relations, preservation of relational converse and diagonal relations, and lax preservation of composition ($LR \circ LS \subseteq L(R \circ S)$). The monotone neighbourhood functor and its polyadic variants have diagonal-preserving lax liftings, and diagonal-preserving lax liftings are easily seen to be inherited along products and subfunctors, so that every functor that has a separating set of monotone predicate liftings has a diagonal-preserving lax lifting. Conversely, every finitary functor that has a diagonal-preserving lax lifting has a separating set of monotone predicate liftings, the so-called Moss liftings [20]. A lax lifting induces a modal logic with a slightly non-standard modality ∇ that generalizes Moss' modality for weak-pullback-preserving functors [23]; for functors that preserve finite sets, the ∇-modality and the predicate-lifting based modalities are however mutually intertranslatable [20], essentially by dint of the fact that both are separating. Summing up, for a functor that preserves finite sets, a diagonal-preserving lax lifting exists iff a separating (finite) set of monotone predicate lifting exists, and the induced logics are essentially the same.

Marti [20] shows that the logic of a diagonal-preserving lax lifting L for T has uniform interpolation if T preserves finite sets and L is *quasifunctorial*, i.e. satisfies $LS \circ LR = L(S \circ R) \cap (\mathsf{dom}(LR) \times \mathsf{rg}(LS))$ where $\mathsf{dom}(LR) = \{t \mid \exists t.\, (s,t) \in LR\}$ and $\mathsf{rg}(LS) = \{t \mid \exists s.\, (s,t) \in LS\}$. We recall again that our reduction of uniform interpolation to one-step interpolation holds also in cases where either separation or monotonicity fails, such as coalition logic / alternating-time logic and neighbourhood logic, respectively. Also, we have seen examples (Abelian-group-weighted functors) where there is no monotone separating set of predicate liftings but we nevertheless obtain uniform interpolation from preservation of surjective weak pullbacks.

6 Conclusions

We have given sufficient criteria for a rank-1 modal logic (with finitely many modalities), i.e. a coalgebraic modal logic, to have uniform interpolation: In the general case, we have established

a reduction to the one-step logic; and in the case where the modalities are separating, we have given a simple semantic criterion, namely preservation of (finite) surjective weak pullbacks, which in the monotone case is in fact also necessary for one-step interpolation. We have thus reproved uniform interpolation for the relational modal logics K and KD and for monotone (neighbourhood) modal logic, and newly established uniform interpolation for coalition logic, neighbourhood logic (i.e. classical modal logic), and various logics of finite-monoid-weighted transition systems. All proofs are entirely semantic; we leave a proof-theoretic treatment, in generalization of tentative results based on cut-free sequent systems [28], for future work. In particular, such a treatment will hopefully lead to practically feasible algorithms for the computation of interpolants. Further open issues concern the question of how our results relate to definability of bisimulation quantifiers [30, 38], and of course the development of generic criteria for interpolation in the presence of infinitely many modalities.

Acknowledgements. The authors wish to thank Tadeusz Litak and Sebastian Enqvist for helpful discussions. Erwin R. Catesbeiana has provided valuable hints on interpolating between unsatisfiable formulas.

References

1 Jiří Adámek, Horst Herrlich, and George Strecker. *Abstract and Concrete Categories*. Wiley Interscience, 1990. Republished in *Reprints in Theor. Appl. Cat.* 17 (2006).

2 Michael Barr. Terminal coalgebras in well-founded set theory. *Theoret. Comput. Sci.*, 114:299–315, 1993.

3 Marta Bílková. Uniform interpolation and propositional quantifiers in modal logics. *Stud. Log.*, 85:1–31, 2007.

4 Brian Chellas. *Modal Logic*. Cambridge University Press, 1980.

5 William Craig. Linear reasoning. A new form of the Herbrand-Gentzen theorem. *J. Symb. Log.*, 22:250–268, 1957.

6 Giovanna D'Agostino and Marco Hollenberg. Logical questions concerning the μ-calculus: Interpolation, Lyndon and Łoś-Tarski. *J. Symb. Log.*, 65:310–332, 2000.

7 Stéphane Demri and Denis Lugiez. Complexity of modal logics with Presburger constraints. *J. Appl. Log.*, 8:233–252, 2010.

8 Dov Gabbay. Craig's interpolation theorem for modal logics. In *Conference in Mathematical Logic—London'70*, volume 255 of *LNM*, pages 111–127. Springer, 1972.

9 Dov Gabbay. Craig interpolation theorem for intuitionistic logic and extensions Part III. *J. Symb. Log.*, 42:269–271, 1977.

10 Silvio Ghilardi. An algebraic theory of normal forms. *Ann. Pure Appl. Log.*, 71:189–245, 1995.

11 Silvio Ghilardi and Marek Zawadowski. Undefinability of propositional quantifiers in the modal system S4. *Stud. Log.*, 55:259–271, 1995.

12 H. Peter Gumm and Tobias Schröder. Coalgebraic structure from weak limit preserving functors. In *Coalgebraic Methods in Computer Science, CMCS 2000*, volume 33 of *ENTCS*, pages 111–131. Elsevier, 2000.

13 H. Peter Gumm and Tobias Schröder. Monoid-labeled transition systems. In *Coalgebraic Methods in Computer Science, CMCS 2001*, volume 44 of *ENTCS*, pages 185–204. Elsevier, 2001.

14 Helle Hansen and Clemens Kupke. A coalgebraic perspective on monotone modal logic. In *Coalgebraic Methods in Computer Science, CMCS 2004*, volume 106 of *ENTCS*, pages 121–143. Elsevier, 2004.

15 Helle Hansen, Clemens Kupke, and Eric Pacuit. Neighbourhood structures: Bisimilarity and basic model theory. *Log. Meth. Comput. Sci.*, 5, 2009.

16 Leon Henkin. An extension of the Craig-Lyndon interpolation theorem. *J. Symb. Log.*, 28:201–216, 1963.

17 Dexter Kozen. Results on the propositional μ-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.

18 Alexander Kurz and Raul Leal. Modalities in the Stone age: A comparison of coalgebraic logics. *Theor. Comput. Sci.*, 430:88–116, 2012. URL: `http://dx.doi.org/10.1016/j.tcs.2012.03.027`, `doi:10.1016/j.tcs.2012.03.027`.

19 Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In *International Joint Conference on Artificial Intelligence, IJCAI 2011*, pages 989–995. IJCAI/AAAI, 2011.

20 Johannes Marti. Relation liftings in coalgebraic modal logic. Master's thesis, Universiteit van Amsterdam, 2011.

21 Johannes Marti, Fatemeh Seifan, and Yde Venema. Uniform interpolation for coalgebraic fixpoint logic. In *Algebra and Coalgebra in Computer Science, CALCO 2015*, volume 35 of *LIPIcs*, pages 238–252. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

22 Johannes Marti and Yde Venema. Lax extensions of coalgebra functors and their logic. *J. Comput. Syst. Sci.*, 81:880–900, 2015.

23 Larry Moss. Coalgebraic logic. *Ann. Pure Appl. Logic*, 96:277–317, 1999.

24 Rob Myers, Dirk Pattinson, and Lutz Schröder. Coalgebraic hybrid logic. In *Foundations of Software Science and Computation Structures, FoSSaCS 2009*, volume 5504 of *LNCS*, pages 137–151. Springer, 2009.

25 Dirk Pattinson. Coalgebraic modal logic: Soundness, completeness and decidability of local consequence. *Theoret. Comput. Sci.*, 309:177–193, 2003.

26 Dirk Pattinson. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame J. Formal Log.*, 45:19–33, 2004.

27 Dirk Pattinson. The logic of exact covers: Completeness and uniform interpolation. In *Logic in Computer Science, LICS 2013*, pages 418–427. IEEE, 2013.

28 Dirk Pattinson and Lutz Schröder. Cut elimination in coalgebraic logics. *Inf. Comput.*, 208:1447–1468, 2010.

29 Marc Pauly. A modal logic for coalitional power in games. *J. Log. Comput.*, 12:149–166, 2002.

30 Andrew Pitts. On an interpretation of second order quantification in first order intuitionistic propositional logic. *J. Symb. Log.*, 57:33–52, 1992.

31 Jan Rutten. Universal coalgebra: A theory of systems. *Theor. Comput. Sci.*, 249:3–80, 2000.

32 Luigi Santocanale and Yde Venema. Uniform interpolation for monotone modal logic. In *Advances in Modal Logic, AiML 2010*, pages 350–370. College Publications, 2010.

33 Lutz Schröder. A finite model construction for coalgebraic modal logic. *J. Log. Algebr. Prog.*, 73:97–110, 2007.

34 Lutz Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. *Theor. Comput. Sci.*, 390:230–247, 2008.

35 Lutz Schröder and Dirk Pattinson. Strong completeness of coalgebraic modal logics. In *Symposium on Theoretical Aspects of Computer Science, STACS 2009*, volume 3 of *LIPIcs*, pages 673–684. Schloss Dagstuhl – Leibniz-Center of Informatics, 2009.

36 Lutz Schröder and Dirk Pattinson. Rank-1 modal logics are coalgebraic. *J. Log. Comput.*, 20(5):1113–1147, 2010.

37 Daniele Turi. *Functorial operational semantics and its denotational dual.* PhD thesis, Vrije Universiteit, Amsterdam, 1996.

38 Albert Visser. Uniform interpolation and layered bisimulation. In *Gödel '96 (Brno, 1996)*, volume 6 of *Lect. Notes Log.*, pages 139–164. Springer, 1996.

39 Kewen Wang, Zhe Wang, Rodney Topor, Jeff Pan, and Grigoris Antoniou. Concept and role forgetting in $\mathcal{ALC}$-ontologies. In *The Semantic Web, ISWC 2009*, volume 5823 of *LNCS*. Springer, 2009.

Termination in Convex Sets of Distributions*

Ana Sokolova[1] **and Harald Woracek**[2]

1 **University of Salzburg, Austria**
ana.sokolova@cs.uni-salzburg.at
2 **TU Vienna, Austria**
harald.woracek@tuwien.ac.at

Abstract

Convex algebras, also called (semi)convex sets, are at the heart of modelling probabilistic systems including probabilistic automata. Abstractly, they are the Eilenberg-Moore algebras of the finitely supported distribution monad. Concretely, they have been studied for decades within algebra and convex geometry.

In this paper we study the problem of extending a convex algebra by a single point. Such extensions enable the modelling of termination in probabilistic systems. We provide a full description of all possible extensions for a particular class of convex algebras: For a fixed convex subset D of a vector space satisfying additional technical condition, we consider the algebra of convex subsets of D. This class contains the convex algebras of convex subsets of distributions, modelling (nondeterministic) probabilistic automata. We also provide a full description of all possible extensions for the class of free convex algebras, modelling fully probabilistic systems. Finally, we show that there is a unique functorial extension, the so-called black-hole extension.

1998 ACM Subject Classification F.3 Logics and Meanings of Programs, G.3 Probability and Statistics, F.1.2 Modes of Computation, D.2.4 Software/Program verification

Keywords and phrases convex algebra, one-point extensions, convex powerset monad

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.22

1 Introduction

In this paper we study the question of how to extend a convex algebra by a single element. Convex algebras have been studied for many decades in the context of algebra, vector spaces, and convex geometry, see e.g. [32, 11, 13] and from a categorical viewpoint, see e.g. [12, 33, 24, 22, 3]. Recently they have attracted more attention in computer science as well, see e.g. [9, 15, 17]. One reason is that probability distributions, the main ingredient for modelling various probabilistic systems, see e.g. [34, 1, 30], have a natural convex algebra structure. Even more than that, the set of finitely supported distributions over a set S carries the free convex algebra over S. As a consequence, on the concrete side, convexity has notably appeared in the semantics of probabilistic systems, in particular probabilistic automata [27, 26, 19, 14]. One particularly interesting development in the last decade in the theory of probabilistic systems is to consider probabilistic automata as transformers of *belief states*, i.e., probability distributions over states, resulting in semantics on distributions, see [14, 4, 5, 7, 8, 6, 21] to name a few. Convexity is inherent to this modelling and the resulting semantics that we call *distribution bisimilarity*.

* Full version http://www.asc.tuwien.ac.at/preprint/2017/asc08x2017.pdf.

Additionally, on the abstract side, coalgebras over (categories of) algebras have attracted significant attention [29, 16]. They make explicit the algebraic structure that is present in (the states of) transition systems and allow for its utilisation in the notion of semantics. For coalgebras over convex algebras, the most important observation is that convex algebras are the Eilenberg-Moore algebras of the finitely supported distribution monad [33, 9, 10, 15]. The first author, with coauthors, has recently studied the abstract coalgebraic foundation of probabilistic automata as coalgebras over convex algebras in [2], by providing suitable functors and monads on the category $\mathcal{EM}(\mathcal{D}_f)$ of Eilenberg-Moore algebras that model probabilistic automata. As a result, one gets a neat generic treatment and understanding of distribution bisimilarity.

One contribution of [2] is identifying a convex-powerset monad $\mathcal{P}_\mathrm{c}$ on $\mathcal{EM}(\mathcal{D}_f)$ that together with a constant-exponent functor can be used to model probabilistic automata as coalgebras over $\mathcal{EM}(\mathcal{D}_f)$. However, the convex-powerset monad allows only for *nonempty* convex subsets, and hence there is no notion of termination. As a consequence, one can only model *input enabled* probabilistic automata. Hence, the question arises of how to add termination. One obvious way is to add termination that rules over any other behaviour: Consider a probabilistic automaton with two states s and t; A distribution $ps+\bar{p}t$ over states s and t terminates if and only if one of the states terminates. We refer to this approach as *black-hole* termination. Several distribution-bisimilarity semantics in the literature disagree on the treatment of termination, see e.g. the discussion in [14] as well as [7, 8, 6, 4]. Understanding termination in probabilistic automata as transformers of belief states is the motivation for this work. On the level of convex algebras, termination amounts to the question of extending a convex algebra by a single element.

Stated algebraically, the questions we address in this paper are:

1. Given a convex algebra $\mathbb{X}$, is it possible to extend it by a single point?
2. If yes, what are all possible extensions?
3. Which extensions are functorial, i.e., provide a functor on $\mathcal{EM}(\mathcal{D}_f)$ that could then be used for modelling probabilistic automata as coalgebras over $\mathcal{EM}(\mathcal{D}_f)$?

Observe that extensions by a single point are different from the coproduct $\mathbb{X} + 1$ in $\mathcal{EM}(\mathcal{D}_f)$; the coproduct was concretely described in [17, Lemma 4], and it has a much larger carrier than the set $X + 1$.

Despite a large body of work on convex algebras, to the best of our knowledge, the problem of extending a convex algebra by a single element has not been studied, except for the black-hole extension mentioned above, see [12].

We answer the stated questions, and in particular our answers and main results are:

1. Yes, it is possible and there are many possible extensions in general. One of them is the mentioned black-hole extension.
2. We describe all possible extensions for the free convex algebra $\mathbb{D}_S$ of finitely supported probability distributions over a set S, see Theorem 16 in Section 5. Furthermore, we describe all possible extensions of an algebra $\mathcal{P}_\mathrm{c}\mathbb{D}$ for $\mathbb{D}$ being a convex subset of a vector space, satisfying a boundedness condition, see Theorem 29 in Section 6. As $\mathbb{D}_S$ is a particular subset of a vector space, we get a description of all possible extensions of $\mathcal{P}_\mathrm{c}\mathbb{D}_S$ which is exactly what is needed to understand termination in convex sets of distributions.
3. We prove that only the black-hole extension is functorial, see Theorem 18 in Section 5.

In addition, we provide many smaller results, observations, and examples that add to the vast knowledge on convex algebras.

We mention that reading our results and proofs in detail does not require any prior knowledge beyond basics of algebra, with two notable exceptions: (1) We do use some topological and geometric arguments in the appendix in order to prove claims for the construction of some of our examples; (2) We add small remarks about coalgebras and categories as we already did in this introduction, assuming that readers are familiar with these basic notions (or will otherwise ignore the remarks made).

2 Convex Algebras

By $\mathcal{C}$ we denote the signature of convex algebras

$$\mathcal{C} = \{(p_i)_{i=0}^n \mid n \in \mathbb{N}, p_i \in [0,1], \sum_{i=0}^n p_i = 1\}.$$

Intuitively, the $(n+1)$-ary operation symbol $(p_i)_{i=0}^n$ will be interpreted by a convex combination with coefficients p_i for $i = 0, \dots, n$. For a real number $p \in [0,1]$ we set $\bar{p} = 1 - p$.

▶ **Definition 1.** A *convex algebra* $\mathbb{X}$ is an algebra with signature $\mathcal{C}$, i.e., a set X together with an operation $\sum_{i=0}^n p_i(-)_i$ for each operational symbol $(p_i)_{i=0}^n \in \mathcal{C}$, such that the following two axioms hold:

- Projection: $\sum_{i=0}^n p_i x_i = x_j$ if $p_j = 1$.
- Barycenter: $\sum_{i=0}^n p_i \left(\sum_{j=0}^m q_{i,j} x_j\right) = \sum_{j=0}^m \left(\sum_{i=0}^n p_i q_{i,j}\right) x_j$.

Convex algebras are the Eilenberg-Moore algebras of the finitely-supported distribution monad $\mathcal{D}_f$ on **Sets**, cf. [33, 4.1.3] and [28], see also [9, 10] or [15, Theorem 4] where a concrete and simple proof is given. A convex algebra homomorphism is a morphism in the Eilenberg-Moore category $\mathcal{EM}(\mathcal{D}_f)$. Concretely, a convex algebra homomorphism h from $\mathbb{X}$ to $\mathbb{Y}$ is a *convex* (synonymously, *affine*) map, i.e., $h\colon X \to Y$ with the property $h\left(\sum_{i=0}^n p_i x_i\right) = \sum_{i=0}^n p_i h(x_i)$.

▶ Remark 2. Let $\mathbb{X}$ be a convex algebra. Then (for $p_n \neq 1$)

$$\sum_{i=0}^n p_i x_i = \overline{p_n} \left(\sum_{j=0}^{n-1} \frac{p_j}{\overline{p_n}} x_j\right) + p_n x_n. \tag{1}$$

Hence, an $(n+1)$-ary convex combination can be written as a binary convex combination using an n-ary convex combination. As a consequence, if X is a set that carries two convex algebras $\mathbb{X}_1$ and $\mathbb{X}_2$ with operations $\sum_{i=0}^n p_i(-)_i$ and $\bigoplus_{i=0}^n p_i(-)_i$, respectively (and binary versions $+$ and $\oplus$, respectively) such that $px + \bar{p}y = px \oplus \bar{p}y$ for all p, x, y, then $\mathbb{X}_1 = \mathbb{X}_2$.

One can also see Equation (1) as a definition – the classical definition of Stone [32, Definition 1]. We make the connection explicit with the next proposition.

▶ **Proposition 3.** *Let X be a set with binary operations $px + \bar{p}y$ for $x, y \in X$ and $p \in (0,1)$. Assume*

- *Idempotence: $px + \bar{p}x = x$ for all $x \in X, p \in (0,1)$.*
- *Parametric commutativity: $px + \bar{p}y = \bar{p}y + px$ for all $x, y \in X, p \in (0,1)$.*
- *Parametric associativity: $p(qx + \bar{q}y) + \bar{p}z = pqx + \overline{pq}\left(\frac{p\bar{q}}{\overline{pq}}y + \frac{\bar{p}}{\overline{pq}}z\right)$ for all $x, y, z \in X, p, q, \in (0,1)$.*

Define n-ary convex operations inductively by the projection axiom and the formula (1). Then X becomes a convex algebra.

Proof. By induction, cf. [32, Lemma 1–Lemma 4][1]. ◀

This allows us to focus on binary convex combinations whenever more convenient.

▶ **Definition 4.** Let $\mathbb{X}$ be a convex algebra, and $C \subseteq X$. C is *convex* if it is the carrier of a subalgebra of $\mathbb{X}$, i.e., if $px + \bar{p}y \in C$ for all $x, y \in C$ and $p \in (0,1)$.

▶ **Definition 5.** Let $\mathbb{X}$ be a convex algebra. An element $z \in X$ is $\mathbb{X}$-*cancellable* if

$$\forall x, y \in X.\ \forall p \in (0,1).\ px + \bar{p}z = py + \bar{p}z \Rightarrow x = y.$$

The convex algebra $\mathbb{X}$ is *cancellative* if every element of X is $\mathbb{X}$-cancellable.

▶ **Definition 6.** Let $\mathbb{X}$ be a convex algebra. An element $x \in X$ *adheres to* an element $y \in X$, notation $x \circ\!\!-\!\!\bullet y$, if $px + \bar{p}y = y$ for all $p \in (0,1)$.

Observe that for a cancellative algebra the adherence relation equals the identity relation. The following simple properties of adherence will be needed on multiple occasions.

▶ **Lemma 7.** *Let $\mathbb{X}$ be a convex algebra. The following properties hold.*

1. *For all $x, y \in X$, $x \circ\!\!-\!\!\bullet y$ if and only if $px + \bar{p}y = y$ for some $p \in (0,1)$.*
2. *The adherence relation is reflexive and convex.*
3. *For all $x, y \in X$, if $x \circ\!\!-\!\!\bullet y$ then $pz + \bar{p}x \circ\!\!-\!\!\bullet pz + \bar{p}y$ for all $z \in X$ and $p \in (0,1)$.*
4. *If z is $\mathbb{X}$-cancellable, then for all $x, y \in X$ and $p \in (0,1)$*

$$pz + \bar{p}x \circ\!\!-\!\!\bullet pz + \bar{p}y \ \Rightarrow\ x \circ\!\!-\!\!\bullet y.$$

Proof.

1. Let $x, y \in X$. Consider the map $\varphi\colon [0,1] \to \mathbb{X}$ defined by $\varphi(p) = px + \bar{p}y$. Easy calculations show that

$$(qp + \bar{q}r)x + \overline{(qp + \bar{q}r)}y = q(px + \bar{p}y) + \bar{q}(rx + \bar{r}y), \tag{2}$$

   showing that φ is convex. The implication $\Rightarrow$ trivially holds. For the implication $\Leftarrow$ assume that $rx + \bar{r}y = y$ for some $r \in (0,1)$. Then $\varphi(0) = y = \varphi(r)$ showing that the kernel of φ is a congruence of $[0,1]$ which is not the diagonal. By [11, Lemma 3.2], φ is constant on $(0,1)$. This shows that for all $p \in (0,1)$, $px + \bar{p}y = y$ and hence $x \circ\!\!-\!\!\bullet y$.
2. Reflexivity is a direct consequence of idempotence. Let $x, y, u, v \in X$ and assume $x \circ\!\!-\!\!\bullet y$ and $u \circ\!\!-\!\!\bullet v$. Then

$$q(px + \bar{p}u) + \bar{q}(py + \bar{p}v) = p(qx + \bar{q}y) + \bar{p}(qu + \bar{q}v) = py + \bar{p}v.$$

3. Direct consequence of reflexivity and convexity of adherence.
4. Assume $pz + \bar{p}x \circ\!\!-\!\!\bullet pz + \bar{p}y$ and z is $\mathbb{X}$-cancellable. Let $q \in (0,1)$. Then

$$pz + \bar{p}y = q(pz + \bar{p}x) + \bar{q}(pz + \bar{p}y) = pz + \bar{p}(qx + \bar{q}y)$$

   implies $qx + \bar{q}y = y$, after cancelling z. Hence $x \circ\!\!-\!\!\bullet y$. ◀

[1] Stone's cancellation Postulate V is not used in his Lemma 1–Lemma 4.

▶ **Example 8.** Here are some examples of convex algebras of interest in this paper:

1. Let $\mathbb{V}$ be a vector space over $\mathbb{R}$ and $X \subseteq V$ a convex subset. Then X with the operations inherited from $\mathbb{V}$ is a cancellative convex algebra $\mathbb{X}$. Conversely, every cancellative convex algebra is isomorphic to a convex subset of a vector space, cf. [32, Theorem 2][2].
2. In particular, we consider the vector space $\ell^1(S)$ for a set S. Recall, $\ell^1(S) = \{(r_s)_{s\in S} \mid r_s \in \mathbb{R}, \sum_{s\in S} |r_s| < \infty\}$ with the norm $\|(r_s)_{s\in S}\|_1 = \sum_{s\in S} |r_s|$. The set $\mathcal{D}_f S$ of finitely supported probability distributions over S forms a convex subset of $\ell^1(S)$ and hence a cancellative convex algebra $\mathbb{D}_S$. It is in fact a well-known convex algebra, the free convex algebra generated by S, cf. [20, Lemma 1].
3. Given a convex algebra $\mathbb{X}$, $\mathcal{P}_c X$ is the set of nonempty[3] convex subsets of X, i.e., carriers of subalgebras of $\mathbb{X}$. $\mathcal{P}_c X$ forms a convex algebra with the pointwise operations: $pA + \bar{p}B = \{pa + \bar{p}b \mid a \in A, b \in B\}$. We write $\mathcal{P}_c\mathbb{X}$ for this algebra. We note that $\mathcal{P}_c$ is a monad on $\mathcal{EM}(\mathcal{D}_f)$ as shown in [2]. On morphisms, $\mathcal{P}_c$ acts as the powerset monad. The original algebra $\mathbb{X}$ embeds in $\mathcal{P}_c\mathbb{X}$ via the unit of the monad $\eta\colon x \mapsto \{x\}$.
   For convex subsets of a finite dimensional vector space, the pointwise operations are known as Minkowski addition and are a basic construction in convex geometry, cf. [25]. The algebra $\mathcal{P}_c\mathbb{X}$ is in general not cancellative and has a nontrivial adherence relation, cf. [12, Example 6.3]. However it contains cancellative elements: It is easy to check that for each $\mathbb{X}$-cancellable element x the element $\{x\}$ is $\mathcal{P}_c\mathbb{X}$-cancellable.
4. The motivating example for this work is the convex algebra $\mathcal{P}_c\mathbb{D}_S$ of convex subsets of distributions over a set S.
5. Let X be the carrier of a meet-semilattice and define $px + \bar{p}y = x \wedge y$ for $x, y \in X$ and $p \in (0, 1)$. Then X becomes a convex algebra $\mathbb{X}$ with these operations, cf. [20, §4.5]. This algebra is not cancellative, in fact $\multimap\!\bullet = \{(x, y) \mid x \geq y\}$. For the categorically minded, we remark that behind this construction is the support monad map from $\mathcal{D}_f$ to $\mathcal{P}_f$, the finite powerset monad, and semilattices are the Eilenberg-Moore algebras of $\mathcal{P}_f$.

We now present a construction that provides a beautiful way of constructing convex algebras out of existing ones.

▶ **Example 9.** The *semilattice construction*, cf. [12, p.22f]: Let S be the carrier of a meet-semilattice and let $(\mathbb{X}_s)_{s\in S}$ be an S-indexed family of convex algebras. Moreover, let $(f_s^t)_{s,t\in S}$ be a family of convex algebra homomorphisms $f_s^t\colon \mathbb{X}_t \to \mathbb{X}_s$ that satisfy $f_s^t \circ f_t^u = f_s^u$ for all $s \leq t \leq u$, and $f_s^s = \mathrm{id}_{X_s}$ for all $s \in S$. Let X be the disjoint union of all X_s for $s \in S$. Then X becomes a convex algebra $\mathbb{X}$ with operations defined by $px + \bar{p}y = pf_{s\wedge t}^s(x) + \bar{p}f_{s\wedge t}^t(y)$ for $x \in X_s$, $y \in X_t$, and $p \in (0, 1)$. The algebra $\mathbb{X}$ obtained in this way is the direct limit of the diagram formed by the algebras $\mathbb{X}_s$ and the maps f_s^t.

▶ **Definition 10.** Let $\mathbb{X}$ be a convex algebra, and $P, Q \subseteq X$.

- P is an *ideal* if $\forall x \in P.\ \forall y \in X.\ \forall p \in (0, 1).\ px + \bar{p}y \in P$.
- P is a *prime ideal* if it is an ideal and its complement $X \setminus P$ is convex.
- Q is an *extremal set*[4] if $px + \bar{p}y \in Q \Rightarrow x, y \in Q$ for all $x, y \in X, p \in (0, 1)$.
- $z \in X$ is an *extremal point* if $\{z\}$ is an extremal set. Explicitly: z is an extremal point if whenever $px + \bar{p}y = z$ for $x, y \in X, p \in (0, 1)$, it follows that $x = y = z$. The set of all extremal points of $\mathbb{X}$ is denoted as $\mathrm{Ext}\,\mathbb{X}$.[5]

[2] Proved independently in [18, Satz 3].
[3] Non-emptiness is necessary for the projection axiom to hold.
[4] In [15, Definition 7] extremal sets are called *filters*.
[5] The construction Ext is not functorial.

▶ **Lemma 11.** *Let $\mathbb{X}$ be a convex algebra, and $P \subseteq X$. Then P is an ideal if and only if $X \setminus P$ is an extremal set.*

Proof. Assume P is an ideal. Let $x, y \in X, p \in (0,1)$ such that $px + \bar{p}y \in X \setminus P$. If $x \in P$ or $y \in P$, then since P is an ideal also $px + \bar{p}y \in P$, a contradiction. Hence $x, y \in X \setminus P$.

For the converse, assume $X \setminus P$ is extremal and let $x \in P, y \in X, p \in (0,1)$. If $px + \bar{p}y \notin P$, then $px + \bar{p}y \in X \setminus P$ which implies $x, y \in X \setminus P$, a contradiction. Hence, $px + \bar{p}y \in P$. ◀

3 The Problem and Some Example Solutions

Let $\mathbb{X}$ be a convex algebra. Can one extend it for one element to a convex algebra $\mathbb{X}_*$ with carrier $X \cup \{*\}$ where $* \notin X$? If yes, what are all possible extensions?

We will show that an arbitrary convex algebra $\mathbb{X}$ can be extended in many ways, and describe all possible ways of extending $\mathbb{X} = \mathbb{D}_S$ and $\mathbb{X} = \mathcal{P}_c\mathbb{D}_S$.

First, we provide four examples of extensions, two of which are instances of the semilattice construction from Example 9.

▶ **Example 12.** Let $\mathbb{X}$ be a convex algebra and $* \notin X$. We denote the operations of $\mathbb{X}$ as before by $p(-) + \bar{p}(-)$. In each of the examples we construct a convex algebra $\mathbb{X}_*$ with operations denoted by $p(-) \oplus \bar{p}(-)$ satisfying $p(-) \oplus \bar{p}(-) = p(-) + \bar{p}(-)$, $x, y \in X$, $p \in [0,1]$.

1. Black-hole behaviour, cf. [12, Example 6.1]: In this example, $*$ behaves like a black hole and swallows everything in the sense that $x \multimap\!\bullet *$ for all $x \in X$. To be precise, consider the semilattice $S = \{0,1\}$ with $0 \leq 1$. Let $\mathbb{X}_0$ be the trivial convex algebra with $X_0 = \{*\}$ and $\mathbb{X}_1 = \mathbb{X}$. Let $f_0^1 : \mathbb{X}_1 \to \mathbb{X}_0$ be the unique homomorphism (mapping everything to $*$). Then the semilattice construction gives us a convex algebra $\mathbb{X}_*$ with the property

$$px \oplus \bar{p}y = \begin{cases} px + \bar{p}y\,, & x, y \in X, \\ *\qquad\quad\;, & x = * \text{ or } y = *. \end{cases} \tag{3}$$

2. Imitating behaviour: In this example, $*$ imitates the behaviour of a given element $w \in X$. Consider again the semilattice $S = \{0,1\}$ with $0 \leq 1$. Let $\mathbb{X}_0 = \mathbb{X}$ and $\mathbb{X}_1$ be the trivial convex algebra with $X_1 = \{*\}$. Let $f_0^1 : \mathbb{X}_1 \to \mathbb{X}_0$ be the homomorphism mapping $*$ to w. Then the semilattice construction gives us a convex algebra $\mathbb{X}_*$ with the property

$$px \oplus \bar{p}y = \begin{cases} px + \bar{p}y\,, & x, y \in X, \\ px + \bar{p}w\,, & x \in X, y = *, \\ pw + \bar{p}y\,, & x = *, y \in X, \\ *\qquad\quad\;, & x = y = *. \end{cases} \tag{4}$$

3. Imitating an outer element: Assume we are given a convex algebra $\mathbb{Y}$ which contains $\mathbb{X}$ as a subalgebra. Let $w \in Y \setminus X$ be such that $X \cup \{w\}$ is convex. Then we obtain an extension $\mathbb{X}_*$ by identifying $X \cup \{*\}$ with $X \cup \{w\}$ via $x \mapsto x$ for $x \in X$ and $* \mapsto w$. We say that $*$ imitates the outer element w.

   This way of defining extensions is of course trivial, but it is useful in presence of a natural larger algebra. For example, we will apply it when D is a convex subset of a vector space $\mathbb{V}$, $\mathbb{X} = \mathcal{P}_c\mathbb{D}$, and $\mathbb{Y} = \mathcal{P}_c\mathbb{V}$.

4. Mixed behaviour: Let w be an extremal point of $\mathbb{X}$. In this example, $*$ imitates $w \in X$ on $X \setminus \{w\}$ and swallows w. That is, setting

$$px \oplus \bar{p}y = \begin{cases} px + \bar{p}y\ , & x, y \in X, \\ px + \bar{p}w\ , & x \in X \setminus \{w\}, y = *, \\ pw + \bar{p}y\ , & x = *, y \in X \setminus \{w\}, \\ *\qquad\ , & \text{otherwise.} \end{cases} \tag{5}$$

provides an extension $\mathbb{X}_*$. This example is not an instance of the semilattice construction and requires a proof. It will be proven in Section 5 (p.8).

4 Extensions of Convex Algebras - The Prime Ideal

The following two notions provide a crucial characteristic of an extension $\mathbb{X}_*$ for a convex algebra $\mathbb{X}$.

▶ **Definition 13.** Let $\mathbb{X}$ be a convex algebra, and let $\mathbb{X}_*$ be an extension. Then its *set of adherence* $\mathrm{Ad}(\mathbb{X}_*)$ is $\mathrm{Ad}(\mathbb{X}_*) = \{x \in X \mid x \circ\!\!-\!\!\bullet *\}$ and its *prime ideal* is $\mathrm{P}(\mathbb{X}_*) = X \setminus \mathrm{Ad}(\mathbb{X}_*)$.

▶ **Lemma 14.** *Let $\mathbb{X}$ be a convex algebra, and let $\mathbb{X}_*$ be an extension of $\mathbb{X}$. The set $\mathrm{P}(\mathbb{X}_*)$ is indeed a prime ideal of $\mathbb{X}$.*

Proof. Let $x \in \mathrm{P}(\mathbb{X}_*)$, $y \in X$, $p \in (0,1)$. Then

$$q(px + \bar{p}y) + \bar{q}* = \overline{q\bar{p}} \left(\frac{qp}{\overline{q\bar{p}}} x + \frac{\bar{q}}{\overline{q\bar{p}}} * \right) + q\bar{p}y \in X$$

since $y \in X$ and $\frac{qp}{\overline{q\bar{p}}} x + \frac{\bar{q}}{\overline{q\bar{p}}} * \in X$ due to $x \in \mathrm{P}(\mathbb{X}_*)$ and hence $x \notin \mathrm{Ad}(\mathbb{X}_*)$. Therefore, $px + \bar{p}y \in \mathrm{P}(\mathbb{X}_*)$ proving that $\mathrm{P}(\mathbb{X}_*)$ is an ideal in $\mathbb{X}$. By Lemma 7.2 $\mathrm{Ad}(\mathbb{X}_*)$ is convex and hence $\mathrm{P}(\mathbb{X}_*)$ is prime. ◀

The next lemma gives a way to conclude that $*$ imitates an element.

▶ **Lemma 15.** *Let $\mathbb{Y}$ be a convex algebra, $\mathbb{X} \leq \mathbb{Y}$ a subalgebra, and let $\mathbb{X}_*$ be an extension of $\mathbb{X}$. Further, let $z \in \mathrm{P}(\mathbb{X}_*)$ and assume that z is $\mathbb{Y}$-cancellable. If there exist $w \in \mathbb{Y}$ and $q \in (0,1)$ with $qz + \bar{q}* = qz + \bar{q}w$, then $*$ imitates w on $\mathrm{P}(\mathbb{X}_*)$ and $\mathrm{Ad}(\mathbb{X}_*) \subseteq \{x \in X \mid x \circ\!\!-\!\!\bullet w\}$.*

Proof. Let $x \in \mathrm{P}(\mathbb{X}_*)$, $p \in (0,1)$, and set $s = \frac{\bar{p}}{\overline{p\bar{q}}}$. Then $s \in (0,1)$ and $\overline{sq} \cdot p = \bar{s}$, $\overline{sq} \cdot \bar{p} = s\bar{q}$, and we have

$$sqz + \overline{sq}(\underbrace{px + \bar{p}*}_{\in \mathrm{P}(\mathbb{X}_*) \subseteq Y}) = s(qz + \bar{q}*) + \bar{s}x = s(qz + \bar{q}w) + \bar{s}x = sqz + \overline{sq}(\underbrace{px + \bar{p}w}_{\in Y}).$$

Cancelling z yields $px + \bar{p}* = px + \bar{p}w$. We conclude that indeed $*$ imitates w on all of $\mathrm{P}(\mathbb{X}_*)$. Assume now that $x \in \mathrm{Ad}(\mathbb{X}_*)$. Then by Lemma 7.3.

$$pz + \bar{p}x \circ\!\!-\!\!\bullet pz + \bar{p}* = pz + \bar{p}w, \qquad \text{for } p \in (0,1).$$

Again using cancellability of z, it follows that $x \circ\!\!-\!\!\bullet w$ by Lemma 7.4. ◀

5 Extensions of Free Algebras and Functoriality

Let S be a nonempty set and consider the free convex algebra over S. As noted in Example 8.2, this is the algebra $\mathbb{D}_S$ of finitely supported distributions on S. In the next theorem we determine all possible one-point extensions of $\mathbb{D}_S$.

▶ **Theorem 16.** *Let S be a nonempty set and consider the free convex algebra $\mathbb{D}_S$. Extensions $(\mathbb{D}_S)_*$ can be constructed as follows:*
1. *The black-hole behaviour, where $\mathrm{Ad}((\mathbb{D}_S)_*) = \mathcal{D}_f S$.*
2. *Let $w \in \mathcal{D}_f S$, and let $*$ imitate w on all of $\mathcal{D}_f S$.*
3. *Let w be an extremal point of $\mathbb{D}_S$, and let $*$ imitate w on $\mathcal{D}_f S \setminus \{w\}$ and adhere w.*

Every extension $(\mathbb{D}_S)_*$ can be obtained in this way, and each two of these extensions are different.

Note that $w \in \operatorname{Ext} \mathbb{D}_S$ if and only if w is a corner point, in other words, a Dirac measure concentrated at one of the points of S.

The fact that the constructions (1) and (2) give extensions is Example 12.1/2. The construction in (3) is Example 12.4, for which we will now provide evidence. First, we prove a more general statement that we call the gluing lemma, which will be needed later as well. It gives a way to produce extensions with a prescribed set of adherence.

▶ **Lemma 17** (Gluing Lemma). *Let $\mathbb{X}$ be a convex algebra, and $P \subseteq X$ a prime ideal. Assume we have convex operations $p(-) \boxplus \bar{p}(-)$ on $\mathbb{P}_*$ that extend $\mathbb{P}$ (whose operations are inherited from $\mathbb{X}$). Assume further that $\mathrm{Ad}(\mathbb{P}_*) = \emptyset$ and that*

$$px + \bar{p}y \multimap\bullet px \boxplus \bar{p}*, \qquad \text{for } x \in P,\ y \in X \setminus P,\ p \in (0,1). \tag{6}$$

Then the operations $p(-) \oplus \bar{p}(-)$, $p \in (0,1)$, defined as follows extend $\mathbb{X}$ to a convex algebra $\mathbb{X}_*$ with $\mathrm{Ad}(\mathbb{X}_*) = X \setminus P$:

$$px \oplus \bar{p}y = \begin{cases} px + \bar{p}y, & x, y \in X, \\ px \boxplus py, & x = *, y \in P \text{ or } x \in P, y = *, \\ *, & \text{otherwise.} \end{cases}$$

Proof of Example 12.4. Assume we are in the situation of Example 12.4, i.e., $\mathbb{X}$ is a convex algebra and w is an extremal point of $\mathbb{X}$. Set $P = X \setminus \{w\}$, then P is a prime ideal. Further, let $\mathbb{P}_*$ be obtained as in Example 12.3 with $\mathbb{P} \leq \mathbb{X}$ by letting $*$ imitate w. Condition (6) is satisfied with equality, and hence the Gluing Lemma provides $\mathbb{X}_*$. The operations $p(-) \oplus \bar{p}(-)$ obtained in this way coincide with those written in Example 12.4. ◀

Proof of Theorem 16. The uniqueness part is easy to see. First, the action of $*$ determines which case of (1)–(3) occurs since $\mathrm{Ad}((\mathbb{D}_S)_*)$ is $\mathcal{D}_f S$ in case (1), $\emptyset$ in case (2), and $\{w\}$ in case (3). Now uniqueness of the point w in (2) and (3) follows since $\mathbb{D}_S$ is cancellative.

We have to show that every extension occurs in one of the described ways. Hence, let an extension $(\mathbb{D}_S)_*$ be given. If $\mathrm{P}((\mathbb{D}_S)_*) = \emptyset$, case (1) takes place. Assume that $\mathrm{P}((\mathbb{D}_S)_*) \neq \emptyset$ and choose $z \in \mathrm{P}((\mathbb{D}_S)_*)$ and $q \in (0,1)$. Set

$$w = \frac{1}{\bar{q}}\big([qz + \bar{q}*] - qz\big) \in \ell^1(S),$$

then $qz + \bar{q}* = qz + \bar{q}w$ by definition. We apply Lemma 15 with $\mathbb{D}_S \leq \ell^1(S)$ and z, w, q. This yields

$$px + \bar{p}* = px + \bar{p}w, \quad x \in \mathrm{P}((\mathbb{D}_S)_*), p \in (0,1), \tag{7}$$

and $\mathrm{Ad}((\mathbb{D}_S)_*) \subseteq \{x \in \mathcal{D}_f S \mid x \circ\!\!-\!\!\bullet w\} \subseteq \{w\}$.

As a linear combination of two elements of $\mathcal{D}_f S$, the element w is finitely supported. Further, by (7),

$$1 = \frac{1}{\bar{p}}\big(\|pz + \bar{p} * \|_1 - p\|z\|_1\big) \le \|w\|_1 \le \frac{1}{\bar{p}}\big(\|pz + \bar{p} * \|_1 + p\|z\|_1\big) = \frac{1+p}{1-p}$$

for all $p \in (0,1)$, and we see that $\|w\|_1 = 1$. Together, $w \in \mathcal{D}_f S$.

If $\mathrm{P}((\mathbb{D}_S)_*) = \mathcal{D}_f S$, we are in case (2) of the theorem. Otherwise, $\mathrm{P}((\mathbb{D}_S)_*) = (\mathcal{D}_f S)\backslash\{w\}$. This implies that w is an extremal point of $\mathbb{D}_S$, and we are in case (3). ◀

Next we investigate functoriality of one-point extensions. We say that a functor $F\colon \mathcal{EM}(\mathcal{D}_f) \to \mathcal{EM}(\mathcal{D}_f)$ naturally provides a one-point extension, if $\mathbb{X} \le F\mathbb{X}$ and $F\mathbb{X}$ has carrier $X \cup \{*\}$ for $* \notin X$ for every algebra $\mathbb{X}$, and $(Ff)|_X = f$ for every convex map $f\colon \mathbb{X} \to \mathbb{Y}$. The latter property is (literally) a natural property: it says that the family of inclusion maps $\iota_X\colon \mathbb{X} \to F\mathbb{X}$ is a natural transformation of the identity functor to F.

An example of a functor possessing these properties is obtained by the black-hole construction: for an algebra $\mathbb{X}$ let $F\mathbb{X}$ be its black-hole extension, and for a convex map $f : \mathbb{X} \to \mathbb{Y}$ let Ff be the extension of f mapping $*$ (of $F\mathbb{X}$) to $*$ (of $F\mathbb{Y}$).

▶ **Theorem 18.** *Let $F : \mathcal{EM}(\mathcal{D}_f) \to \mathcal{EM}(\mathcal{D}_f)$ be a functor such that for all objects $\mathbb{X}$ and for all morphisms $f : \mathbb{X} \to \mathbb{Y}$*

$$\mathbb{X} \le F\mathbb{X},\ \textit{the carrier of } F\mathbb{X} \textit{ is } X \cup \{*\} \textit{ with } * \notin X, \qquad \begin{array}{ccc} F\mathbb{X} & \xrightarrow{Ff} & F\mathbb{Y} \\ {\scriptstyle \iota_X}\uparrow & & \uparrow{\scriptstyle \iota_Y} \\ \mathbb{X} & \xrightarrow[f]{} & \mathbb{Y} \end{array} \tag{8}$$

Then, for all $\mathbb{X}$, $F\mathbb{X}$ is the black-hole extension, and for all $f\colon \mathbb{X} \to \mathbb{Y}$, Ff is the extension of f mapping $*$ (of $F\mathbb{X}$) to $*$ (of $F\mathbb{Y}$).

We present the proof using two lemmas.

▶ **Lemma 19.** *Assume that $F\colon \mathcal{EM}(\mathcal{D}_f) \to \mathcal{EM}(\mathcal{D}_f)$ satisfies (8), and let $f\colon \mathbb{X} \to \mathbb{Y}$ be a convex map. Then $(Ff)(*) = *$ and $f(\mathrm{P}(F\mathbb{X})) \subseteq \mathrm{P}(F\mathbb{Y})$, $f(\mathrm{Ad}(F\mathbb{X})) \subseteq \mathrm{Ad}(F\mathbb{Y})$.*

Proof. For the proof of $(Ff)(*) = *$, note that $(Ff)^{-1}(\{*\}) \subseteq \{*\}$ since $(Ff)|_X = f$. If f has a right inverse, say $g : \mathbb{Y} \to \mathbb{X}$ with $f \circ g = \mathrm{id}_{\mathbb{Y}}$, then $(Ff)((Fg)(*)) = *$, and hence $(Fg)(*) = *$. In turn also $(Ff)(*) = *$. Now let f be arbitrary. Let $\mathbb{Z}$ be an algebra which has only one element, a final object of $\mathcal{EM}(\mathcal{D}_f)$, and let $h : \mathbb{Y} \to \mathbb{Z}$ be the unique convex map. The map $h \circ f$ has a right inverse, and therefore $(Fh)((Ff)(*)) = (F(h \circ f))(*) = *$. Again, we obtain $(Ff)(*) = *$.

It remains to prove that f maps the respective prime ideals (sets of adherence) into each other. Let $x \in X$ and $p \in (0,1)$. Then

$$pf(x) + \bar{p}* = p(Ff)(x) + \bar{p}(Ff)(*) = (Ff)(px + \bar{p}*) = \begin{cases} f(px + \bar{p}*) \in Y\,, & x \in \mathrm{P}(F\mathbb{X}), \\ (Ff)(*) = * \quad\ \ , & x \in \mathrm{Ad}(F\mathbb{X}). \end{cases} \tag{9}$$

Thus, indeed, $f(x) \in \mathrm{P}(F\mathbb{Y})$ if $x \in \mathrm{P}(F\mathbb{X})$, and $f(x) \in \mathrm{Ad}(F\mathbb{Y})$ if $x \in \mathrm{Ad}(F\mathbb{X})$. ◀

▶ **Lemma 20.** *Assume that $F\colon \mathcal{EM}(\mathcal{D}_f) \to \mathcal{EM}(\mathcal{D}_f)$ satisfies (8), and let S be an infinite set. Then $F\mathbb{D}_S$ is the black-hole extension of $\mathbb{D}_S$.*

Proof. Assume towards a contradiction that $\mathrm{P}(F\mathbb{D}_S) \neq \emptyset$. By Theorem 16 we find $w \in \mathcal{D}_f S$ such that $px + \bar{p}* = px + \bar{p}w$, $x \in \mathrm{P}(F\mathbb{D}_S)$, $p \in (0,1)$. Fix $x \in \mathrm{P}(F\mathbb{D}_S)$ and $p \in (0,1)$, and let $f\colon \mathbb{D}_S \to \mathbb{D}_S$ be an automorphism. Then $f(x) \in \mathrm{P}(F\mathbb{D}_S)$ by Lemma 19, and we can compute

$$pf(x) + \bar{p}w = pf(x) + \bar{p}* \overset{(9)}{=} f(px + \bar{p}*) = f(px + \bar{p}w) = pf(x) + \bar{p}f(w).$$

Cancelling $f(x)$ gives $w = f(w)$. Hence w is a fixpoint of every automorphism.

Since S is infinite, we can choose a point $s_1 \in S$ which lies outside of the support of w. Further, let $s_2 \in S$ be in the support of w, and let $\sigma\colon S \to S$ be the permutation of S which exchanges s_1 and s_2 and leaves all other points fixed. Since $\mathbb{D}_S$ is free with basis S, this permutation extends to an automorphism f of $\mathbb{D}_S$. But now $f(w) \neq w$, a contradiction. ◀

Proof of Theorem 18. The fact that Ff is the extension of f mapping $*$ to $*$ was shown in Lemma 19. It remains to show that, for every algebra $\mathbb{X}$, $F\mathbb{X}$ is the black-hole extension. Given $\mathbb{X}$, choose an infinite set S and a surjective convex map $f : \mathbb{D}_S \to \mathbb{X}$. This is possible since every convex algebra is the image of a free convex algebra, and if $S \supseteq S'$ then there is a surjective homomorphism from $\mathbb{D}_S$ to $\mathbb{D}_{S'}$. Then, by Lemma 19 and Lemma 20, $\mathrm{Ad}(F\mathbb{X}) \supseteq f(\mathrm{Ad}(F\mathbb{D}_S)) = f(\mathbb{D}_S) = \mathbb{X}$. ◀

6 Extensions of $\mathcal{P}_\mathrm{c}\mathbb{D}$

In this section we formulate and prove Theorem 29 where we describe the set of all extensions $(\mathcal{P}_\mathrm{c}\mathbb{D})_*$ for convex algebras $\mathbb{D}$ which are convex subsets of a vector space (equivalently, cancellative) and satisfy a certain linear boundedness condition. Theorem 29 applies in particular to the algebra $\mathbb{D} = \mathbb{D}_S$ of finitely supported distributions over S.

We start with some algebraic preliminaries. First, we recall the notion of linear boundedness, see e.g. [3, Definition 1.1].

▶ **Definition 21.** A convex algebra $\mathbb{X}$ is *linearly bounded*, if every homomorphism of the convex algebra $(0,\infty)$ into $\mathbb{X}$ is constant.

Intuitively, a convex algebra is linearly bounded if it does not contain an infinite ray. A large class of examples of linearly bounded algebras is given by topologically bounded subsets of a topological vector space.

▶ **Example 22.** Let $\mathbb{V}$ be a topological vector space. A subset $D \subseteq V$ is *bounded*, if for every neighbourhood U of 0 there exists $r_0 > 0$ such that $D \subseteq rU$, $r > r_0$. In particular, if $\mathbb{V}$ is a normed space (with a norm denoted by $\|.\|$), then a subset D is bounded in this sense if and only if $\sup_{x\in D} \|x\| < \infty$.

Let $\mathbb{V}$ be a topological vector space. Then for every bounded convex subset D of $\mathbb{V}$, the convex algebra $\mathbb{D}$ is linearly bounded. We could not find an explicit reference for this (intuitive) fact, and hence provide a proof in [31, Appendix B].

▶ Remark 23. Let $\mathbb{V}$ be a vector space over $\mathbb{R}$. Then, for each fixed $w \in \mathbb{V}$ and $t \in \mathbb{R}\setminus\{0\}$, we have the translation map $x \mapsto x + w$ and the scaling map $x \mapsto tx$. They are bijective convex maps on $\mathbb{V}$. Applying $\mathcal{P}_\mathrm{c}$ on these maps gives bijective convex maps on $\mathcal{P}_\mathrm{c}\mathbb{V}$. Moreover, a subset $A \in \mathcal{P}_\mathrm{c}\mathbb{V}$ is linearly bounded if and only if $t(A + w)$ is linearly bounded.

The following observation holds for all cancellative convex algebras $\mathbb{D}$.

▶ **Lemma 24.** *Let $\mathbb{D}$ be a convex algebra and consider $\mathbb{X} = \mathcal{P}_\mathrm{c}\mathbb{D}$. If $\mathbb{D}$ is cancellative, then $A \multimap\!\bullet \{x\} \Rightarrow A = \{x\}$ for all $A \in X$, $x \in D$.*

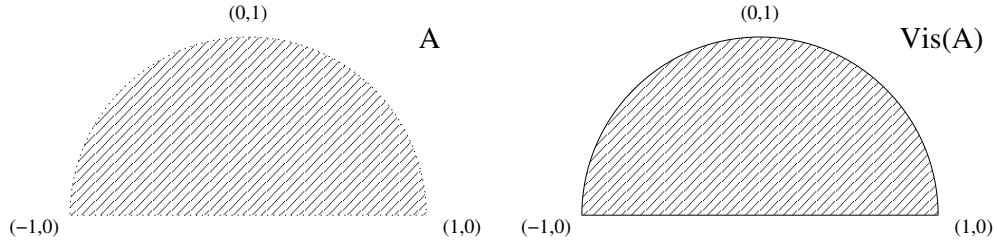


Figure 1 A and $\mathrm{Vis}(A)$.

Proof. Let $a \in A$. Then $pa + \bar{p}x = x = px + \bar{p}x$ which after cancelling with x yields $a = x$. Since A is nonempty, as it belongs to $\mathcal{P}_{\mathrm{c}}\mathbb{D}$, we get $A = \{x\}$. ◀

Under a linear boundedness condition, the roles of A and $\{x\}$ can be exchanged.

▶ **Lemma 25.** *Let $\mathbb{V}$ be a vector space over $\mathbb{R}$, let $A \in \mathcal{P}_{\mathrm{c}}\mathbb{V}$, and assume that $A - A$ is linearly bounded. Then*

$$\bigcap_{p\in[0,1)} \big(p\{x\} + \bar{p}A\big) \subseteq \{x\}, \qquad \text{for } x \in V.$$

In particular, $\{x\} \multimap A \Rightarrow A = \{x\}$ for all $x \in V$.

Proof. Note first that $A - A$ is convex. Let y belong to the intersection. Then $y \in A$ and for each $p \in (0,1)$ we find $a_p \in A$ with $y = px + \bar{p}a_p$. This implies

$$\frac{p}{\bar{p}}(x - y) = y - a_p \in A - A, \quad \text{for } p \in (0,1).$$

Any positive real number t can be represented as $\frac{p}{\bar{p}}$, namely with $p = \frac{t}{1+t} \in (0,1)$. It is easy to check then that $\varphi\colon t \mapsto t(x-y)$ is a convex homomorphism from $(0,\infty)$ to $A - A$. Since $A - A$ is linearly bounded, φ is constant, which further implies $x = y$. ◀

In order to construct extensions where $*$ imitates an outer element, we need the following notion of visibility closure.

▶ **Definition 26.** Let $\mathbb{X}$ be a convex algebra and $A \in \mathcal{P}_{\mathrm{c}}\mathbb{X}$. The *visibility hull* of A is

$$\mathrm{Vis}(A) = \big\{x \in X \mid \forall a \in A.\ \forall p \in (0,1).\ px + \bar{p}a \in A\big\}.$$

The set A is *visibility closed* if $A = \mathrm{Vis}(A)$.

▶ **Example 27.** Let $A \subseteq \mathbb{R}^2$ be the open half-disk $A = \{(t_1,t_2) \in \mathbb{R}^2 \mid t_1^2 + t_2^2 < 1, t_2 > 0\}$. Then $\mathrm{Vis}(A)$ is the closed half disk, shown in Figure 1.

Now consider $B = A \cup \{(0,0)\}$. Then the part of the boundary of B located on the t_1-axis does not belong to $\mathrm{Vis}(B)$, see Figure 2.

Let $\mathbb{V}$ be a vector space over $\mathbb{R}$. The *affine hull* of a subset $A \subseteq V$ is

$$\mathrm{aff}(A) = \{\sum_{i=1}^{n} t_i x_i \mid n \geq 1, x_i \in A, t_i \in \mathbb{R}, \sum_{i=1}^{n} t_i = 1\}.$$

The affine hull of A is the smallest affine subspace of $\mathbb{V}$ containing A, see e.g. [23, p.6].

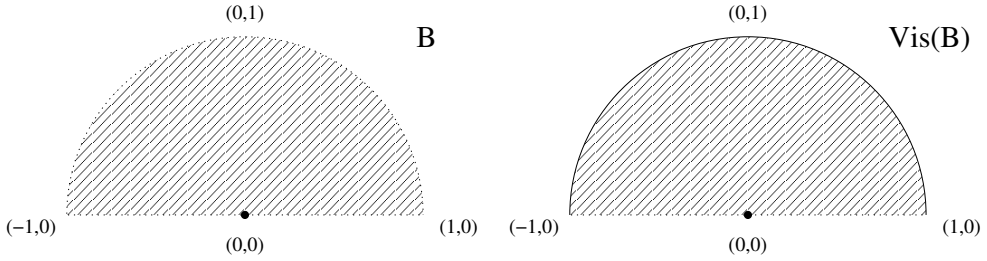


Figure 2 B and $\mathrm{Vis}(B)$.

▶ **Lemma 28.** *Let $\mathbb{V}$ be a vector space over $\mathbb{R}$, and $A \in \mathcal{P}_c\mathbb{V}$. Then*

1. $\mathrm{Vis}(A) = \bigcap_{\substack{a\in A\\ p\in(0,1)}} \frac{1}{p}(A - \bar{p}a) \subseteq \mathrm{aff}(A)$.
2. $\mathrm{Vis}(A)$ *is convex.*
3. $A \subseteq \mathrm{Vis}(A)$ *and* $\mathrm{Vis}(\mathrm{Vis}(A)) = \mathrm{Vis}(A)$.
4. $\mathrm{Vis}(\{z\}) = \{z\}$ *for all* $z \in V$.
5. *If $\mathbb{V}$ is a topological vector space, then* $\mathrm{Vis}(A) \subseteq \overline{A}$. [6]

Proof.

1. We have
$$x \in \mathrm{Vis}(A) \;\Leftrightarrow\; \forall a \in A.\, \forall p \in (0,1).\, px + \bar{p}a \in A \;\Leftrightarrow\; \forall a \in A.\, \forall p \in (0,1).\, x \in \frac{1}{p}(A - \bar{p}a)$$
2. By 1., the set $\mathrm{Vis}(A)$ is the intersection of convex sets.
3. Let $x \in A$. Then $px + \bar{p}a \in A$, $a \in A$, $p \in (0,1)$, since A is convex. Thus $A \subseteq \mathrm{Vis}(A)$. Assume that $x \in \mathrm{Vis}(\mathrm{Vis}(A))$, and let $a \in A$, $p, q \in (0,1)$. Then $px + \bar{p}a \in \mathrm{Vis}(A)$, since $a \in A \subseteq \mathrm{Vis}(A)$, and hence $qpx + \overline{qp}a = q(px + \bar{p}a) + \bar{q}a \in A$. Every number $r \in (0,1)$ can be represented as $r = pq$ with some $p, q \in (0,1)$, and we conclude that $x \in \mathrm{Vis}(A)$.
4. We have $\frac{1}{p}(\{z\} - \bar{p}z) = \{z\}$, $p \in (0,1)$. By 1., $\mathrm{Vis}(\{z\}) = \{z\}$.
5. Let $x \in \mathrm{Vis}(A)$ and $a \in A$. Then $x = \lim_{p\to 1}(px + \bar{p}a) \in \overline{A}$. ◀

The operator $\mathrm{Vis}\colon \mathcal{P}_c\mathbb{V} \to \mathcal{P}_c\mathbb{V}$ is not monotone, as demonstrated in Example 27. Hence, it is not the restriction of a topological closure operator to $\mathcal{P}_c\mathbb{V}$. Still, it is related with topological closures: Let $\mathbb{V}$ be a topological vector space and $A \in \mathcal{P}_c\mathbb{V}$ *relatively closed*, i.e., closed in $\mathrm{aff}(A)$ w.r.t. the subspace topology. Then A is visibility closed (remember footnote 6). This observation shows for example that $\mathrm{Vis}(\mathcal{D}_f S) = \mathcal{D}_f S$. The converse does not hold, as demonstrated by the set $\mathrm{Vis}(B)$ from Example 27.

We can now formulate our description of extensions of $\mathcal{P}_c\mathbb{D}$.

▶ **Theorem 29.** *Let $\mathbb{V}$ be a vector space over $\mathbb{R}$, let D be a convex subset of V with more than one element, and consider the convex algebra $\mathbb{X} = \mathcal{P}_c\mathbb{D}$. Extensions $\mathbb{X}_*$ can be constructed as follows:*

1. *The black-hole behaviour, where* $\mathrm{Ad}(\mathbb{X}_*) = X$.
2. *Let $C \in \mathcal{P}_c(\mathrm{Vis}(D))$, and let $*$ imitate C on all of X.*
3. *Let w be an extremal point of $\mathbb{D}$, and let $*$ imitate $\{w\}$ on $X \setminus \{\{w\}\}$ and adhere $\{w\}$.*

[6] In view of 1., $\mathrm{Vis}(A)$ is contained in the *relative closure* of A — the closure of A in $\mathrm{aff}(A)$ w.r.t. the subspace topology.

4. *Let $C \in \mathcal{P}_{\mathrm{c}}(\mathrm{Vis}(D))$ with at least two elements, assume $\mathrm{conv}\{A \in X \mid A \circ\!\!\not\!\!\bullet C\} \neq X$, and let $I = \mathrm{conv}\{A \in X \mid A \circ\!\!\not\!\!\bullet C\}$. Let $P \neq X$ be a prime ideal in $\mathbb{X}$ with $I \subseteq P$, and let $*$ imitate C on P and adhere $X \setminus P$.*

Assume in addition that $D - D$ is linearly bounded. Then every extension $\mathbb{X}_*$ can be obtained in this way. Each two of these extensions are different: the point w in case (3), the set C in cases (2), (4), and the prime ideal P in case (4), are uniquely determined by $\mathbb{X}_*$.

We are familiar with the constructions (1)–(3) from Example 12 and Theorem 16. That (4) gives extensions follows from the Gluing Lemma, Lemma 17. We give an explicit proof in [31, Appendix C] and illustrative examples in [31, Appendix D].

Assume that $D - D$ is linearly bounded. Our task is to show that every given extension $\mathbb{X}_*$ can be realised as described in (1)–(4) of the theorem, and show uniqueness. The proof relies on the following lemma.

▶ **Lemma 30.** *Assume $\mathbb{X}_*$ is an extension with $\mathrm{P}(\mathbb{X}_*) \neq \emptyset$. Then $\mathrm{Ad}(\mathbb{X}_*)$ contains at most one singleton set.*

Proof. Assume that $\{x\}, \{y\} \in \mathrm{Ad}(\mathbb{X}_*)$, and choose $A \in \mathrm{P}(\mathbb{X}_*)$. Then, for each $p \in (0,1)$, by Lemma 7.3

$$pA + \bar{p}\{x\} \circ\!\!-\!\!\bullet pA + \bar{p}*, \quad pA + \bar{p}\{y\} \circ\!\!-\!\!\bullet pA + \bar{p}\,*\,.$$

Set $C = pA + \bar{p}*$. Then $C \in \mathrm{P}(\mathbb{X}_*)$ and for each $q \in (0,1)$

$$q(pA + \bar{p}\{x\}) + \bar{q}C = C = q(pA + \bar{p}\{y\}) + \bar{q}C.$$

Thus, for each $a \in A, c \in C$ we find $a_1 \in A, c_1 \in C$ with

$$q\bar{p}x + \overline{q\bar{p}}\Big(\frac{qp}{\overline{q\bar{p}}}a + \frac{\bar{q}}{\overline{q\bar{p}}}c\Big) = q\bar{p}y + \overline{q\bar{p}}\Big(\frac{qp}{\overline{q\bar{p}}}a_1 + \frac{\bar{q}}{\overline{q\bar{p}}}c_1\Big),$$

and hence

$$\frac{q\bar{p}}{\overline{q\bar{p}}}(x - y) \in D - D.$$

Any positive real number t can be represented as $\frac{q\bar{p}}{\overline{q\bar{p}}}$ with some $p, q \in (0,1)$, for example use $p = \frac{1}{2t+1}, q = \frac{2t+1}{2t+2}$. Thus $\varphi\colon t \mapsto t(x-y)$ is a homomorphism of $(0,\infty)$ to $D - D$. Since $D - D$ is linearly bounded, φ is constant, and hence $x = y$. ◀

Proof (all $\mathbb{X}_*$ are obtained). Let an extension $\mathbb{X}_*$ of $\mathbb{X}$ be given. If $\mathrm{P}(\mathbb{X}_*) = \emptyset$ then case (1) of the theorem holds. Assume in the following that $\mathrm{P}(\mathbb{X}_*) \neq \emptyset$.

By Lemma 30, $\mathrm{Ad}(\mathbb{X}_*)$ contains at most one singleton set. Since D has more than one element, we find $z \in D$ with $\{z\} \in \mathrm{P}(\mathbb{X}_*)$. Choose $q \in (0,1)$. Then $q\{z\} + \bar{q}* \in \mathrm{P}(\mathbb{X}_*) \subseteq \mathcal{P}_{\mathrm{c}}\mathbb{V}$. We will show that $*$ imitates the convex set

$$C = \frac{1}{\bar{q}}\big([q\{z\} + \bar{q}*] - qz\big) \in \mathcal{P}_{\mathrm{c}}\mathbb{V}.$$

By definition, C satisfies $q\{z\} + \bar{q}* = q\{z\} + \bar{q}C$. Since singletons are $\mathcal{P}_{\mathrm{c}}\mathbb{V}$-cancellable, cf. Example 8.3, the hypothesis of Lemma 15 are fulfilled. We conclude that $*$ imitates C on $\mathrm{P}(\mathbb{X}_*)$ and that $\mathrm{Ad}(\mathbb{X}_*) \subseteq \big\{A \in X \mid A \circ\!\!-\!\!\bullet C\big\}$.

Consider the case that $\mathrm{Ad}(\mathbb{X}_*)$ contains a singleton, say $\{w\} \in \mathrm{Ad}(\mathbb{X}_*)$. Since $C \subseteq \frac{1}{\bar{q}}(D - qz)$, the set $C - C$ is linearly bounded, cf. Remark 23. Lemma 25 implies that

$C = \{w\}$ and Lemma 24 that $\{A \in X \mid A \circ\!\!-\!\!\bullet C\} = \{\{w\}\}$. We see that $\mathrm{P}(\mathbb{X}_*) = X \setminus \{\{w\}\}$ and that $*$ imitates $\{w\}$ on $\mathrm{P}(\mathbb{X}_*)$. Since $X \setminus \{\{w\}\}$ is an ideal in $\mathbb{X}$, also $D \setminus \{w\}$ is an ideal in $\mathbb{D}$, i.e., w is an extremal point of $\mathbb{D}$. Thus $\mathbb{X}_*$ has the form described in case (3).

Consider the case that $\mathrm{Ad}(\mathbb{X}_*)$ contains no singleton. Hence all singletons are in $\mathrm{P}(\mathbb{X}_*)$. Then

$$p\{y\} + \bar{p}C = p\{y\} + \bar{p}* \subseteq X, \qquad \text{for } y \in D,\ p \in (0,1).$$

Thus $C \subseteq \bigcap_{\substack{y \in D \\ p \in (0,1)}} \frac{1}{\bar{p}}(D - py) = \mathrm{Vis}(D)$, by Lemma 28.1. If $\mathrm{Ad}(\mathbb{X}_*) = \emptyset$, case (2) of the theorem holds. Assume that $\mathrm{Ad}(\mathbb{X}_*) \neq \emptyset$. If C contains only one element, say $C = \{w\}$, we would have

$$\emptyset \neq \mathrm{Ad}(\mathbb{X}_*) \subseteq \{A \in X \mid A \circ\!\!-\!\!\bullet \{w\}\} = \{\{w\}\}.$$

From this $\mathrm{Ad}(\mathbb{X}_*) = \{\{w\}\}$, a contradiction. Thus C has at least two elements. Since

$$X \neq \mathrm{P}(\mathbb{X}_*) = X \setminus \mathrm{Ad}(\mathbb{X}_*) \supseteq \{A \in X \mid A \circ\!\!\not\!-\!\!\bullet C\},$$

the convex hull of $\{A \in X \mid A \circ\!\!\not\!-\!\!\bullet C\}$ is not X and case (4) of the theorem holds. ◀

Proof (uniqueness). The uniqueness assertion of the theorem follows since $\mathrm{P}(\mathbb{X}_*)$ always contains singletons by Lemma 30, and singletons are cancellable in $\mathcal{P}_\mathrm{c}\mathbb{V}$. ◀

The following example shows that the linear boundedness condition in Theorem 29 cannot be dropped without admitting other types of constructions. We give the proof in [31, Appendix C].

▶ **Example 31.** Let $\mathbb{V} = \mathbb{R}^2$ and $D = \{(t_1, t_2) \in \mathbb{R}^2 \mid t_2 > 0\} \cup \{(0,0)\}$. Set $P = \{A \in \mathcal{P}_\mathrm{c}\mathbb{D} \mid (0,0) \notin A\}$, then P is a prime ideal of $\mathcal{P}_\mathrm{c}\mathbb{D}$ (see proof in [31, Appendix C]).

Set $C = D \cup \{(t_1, t_2) \in \mathbb{R}^2 \mid t_2 = 0, t_1 > 0\}$. Then $C \subseteq \mathrm{Vis}(D \setminus \{(0,0)\})$, and we can define an extension $\mathbb{P}_*$ by letting $*$ imitate C on all of P. The assumptions of the Gluing Lemma are satisfied (see proof in [31, Appendix C]), and we obtain an extension $(\mathcal{P}_\mathrm{c}\mathbb{D})_*$. This extension is not among the ones listed in Theorem 29, since $C \not\subseteq \mathrm{Vis}(D)$.

Unboundedness of D enters in this example in the way that it enables us to let $*$ imitate a cone. In fact, dropping linear boundedness, one can still show that an extension which is not of type (1)–(4) of the theorem, must be such that $*$ imitates some cone whose apex lies in D. However, we have no description which cones occur that way.

7 Conclusions

We have studied the possibility of extending a convex algebra by a single element. We have proven that many different extensions are possible of which only one gives rise to a functor on $\mathcal{EM}(\mathcal{D}_f)$. We have described all extensions of $\mathbb{D}_S$, the free convex algebra of probability distributions over a set S, and of $\mathcal{P}_\mathrm{c}\mathbb{D}$, the convex algebra of convex subsets of a particular kind of convex subset of a vector space. As a consequence of the latter result, we have described all extensions of $\mathcal{P}_\mathrm{c}\mathbb{D}_S$ used for modelling probabilistic automata.

We expect that the methods developed here can be useful in the study of Eilenberg-Moore algebras for the Giry monad. Detailed investigation is left for future work.

References

1 Falk Bartels, Ana Sokolova, and Erik de Vink. A hierarchy of probabilistic system types. *Theoretical Computer Science*, 327:3–22, 2004.

2 Filippo Bonchi, Alexandra Silva, and Ana Sokolova. The power of convex algebras, 2017. Submitted.

3 Reinhard Börger and Ralf Kemper. A cogenerator for preseparated superconvex spaces. *Appl. Categ. Structures*, 4(4):361–370, 1996.

4 Pablo Samuel Castro, Prakash Panangaden, and Doina Precup. Equivalence relations in fully and partially observable Markov decision processes. In *Proc. IJCAI*, pages 1653–1658, 2009.

5 Silvia Crafa and Francesco Ranzato. A spectrum of behavioral relations over LTSs on probability distributions. In *Proc. CONCUR*, pages 124–139. LNCS 6901, 2011.

6 Yuxin Deng and Matthew Hennessy. On the semantics of Markov automata. *Inf. Comput.*, 222:139–168, 2013.

7 Yuxin Deng, Rob J. van Glabbeek, Matthew Hennessy, and Carroll Morgan. Characterising testing preorders for finite probabilistic processes. *LMCS*, 4(4), 2008.

8 Yuxin Deng, Rob J. van Glabbeek, Matthew Hennessy, and Carroll Morgan. Testing finitary probabilistic processes. In *Proc. CONCUR*, pages 274–288. LNCS 5710, 2009.

9 Ernst-Erich Doberkat. Eilenberg-Moore algebras for stochastic relations. *Inform. and Comput.*, 204(12):1756–1781, 2006.

10 Ernst-Erich Doberkat. Erratum and addendum: Eilenberg-Moore algebras for stochastic relations. *Inform. and Comput.*, 206(12):1476–1484, 2008.

11 Joe Flood. Semiconvex geometry. *J. Austral. Math. Soc. Ser. A*, 30(4):496–510, 1980/81.

12 Tobias Fritz. Convex spaces I: Definition and Examples. arXiv:0903.5522v3 [math.MG].

13 Stanley Gudder and Franklin Schroeck. Generalized convexity. *SIAM J. Math. Anal.*, 11(6):984–1001, 1980.

14 Holger Hermanns, Jan Krcál, and Jan Kretínský. Probabilistic bisimulation: Naturally on distributions. In *Proc. CONCUR*, LNCS 8704, pages 249–265, 2014.

15 Bart Jacobs. Convexity, duality and effects. In *Theoretical computer science*, volume 323 of *IFIP Adv. Inf. Commun. Technol.*, pages 1–19. Springer, 2010.

16 Bart Jacobs, Alexandra Silva, and Ana Sokolova. Trace semantics via determinization. *J. Comput. Syst. Sci.*, 81(5):859–879, 2015.

17 Bart Jacobs, Bas Westerbaan, and Bram Westerbaan. States of convex sets. In *Proc. FOSSACS*, pages 87–101. LNCS 9034, 2015.

18 Hellmuth Kneser. Konvexe Räume. *Arch. Math.*, 3:198–206, 1952.

19 Matteo Mio. Upper-expectation bisimilarity and łukasiewicz μ-calculus. In *Proc. FOSSACS*, pages 335–350. LNCS 8412, 2014.

20 Walter D. Neumann. On the quasivariety of convex subsets of affine spaces. *Arch. Math. (Basel)*, 21:11–16, 1970.

21 Augusto Parma and Roberto Segala. Logical characterizations of bisimulations for discrete probabilistic systems. In *Proc. FOSSACS*, pages 287–301. LNCS 4423, 2007.

22 Dieter Pumplün and Helmut Röhrl. Convexity theories. IV. Klein-Hilbert parts in convex modules. *Appl. Categ. Structures*, 3(2):173–200, 1995.

23 R. Tyrrell Rockafellar. *Convex analysis*. Princeton Mathematical Series, No. 28. Princeton University Press, Princeton, N.J., 1970.

24 Helmut Röhrl. Convexity theories. 0. Foundations. *Appl. Categ. Structures*, 2(1):13–43, 1994.

25 Rolf Schneider. *Convex bodies: the Brunn-Minkowski theory*, volume 44 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1993.

26 Roberto Segala. *Modeling and verification of randomized distributed real-time systems*. PhD thesis, MIT, 1995.

27 Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. In *Proc. CONCUR*, pages 481–496. LNCS 836, 1994.

28 Zbigniew Semadeni. *Monads and their Eilenberg-Moore algebras in functional analysis*. Queen's University, Kingston, Ont., 1973. Queen's Papers in Pure and Applied Mathematics, No. 33.

29 Alexandra Silva, Filippo Bonchi, Marcello Bonsangue, and Jan Rutten. Generalizing the powerset construction, coalgebraically. In *Proc. FSTTCS*, pages 272–283. LIPIcs 8, 2010.

30 Ana Sokolova. Probabilistic systems coalgebraically: A survey. *Theoretical Computer Science*, 412(38):5095–5110, 2011.

31 Ana Sokolova and Harald Woracek. Termination in Convex Sets of Distributions. 25 pp., ASC Report 8, Vienna University of Technology, 2017.
http://www.asc.tuwien.ac.at/preprint/2017/asc08x2017.pdf.

32 Marshall Harvey Stone. Postulates for the barycentric calculus. *Ann. Mat. Pura Appl. (4)*, 29:25–30, 1949.

33 Tadeusz Świrszcz. Monadic functors and convexity. *Bull. Acad. Polon. Sci. Sér. Sci. Math. Astronom. Phys.*, 22:39–42, 1974.

34 Erik de Vink and Jan Rutten. Bisimulation for probabilistic transition systems: a coalgebraic approach. *Theoretical Computer Science*, 221:271–293, 1999.

Precongruences and Parametrized Coinduction for Logics for Behavioral Equivalence

David Sprunger[1] **and Lawrence S. Moss**[*2]

1 **Indiana University, Bloomington, United States**
dasprung@indiana.edu
2 **Indiana University, Bloomington, United States**
lsm@cs.indiana.edu

Abstract

We present a new proof system for equality of terms which present elements of the final coalgebra of a finitary set functor. This is most important when the functor is finitary, and we improve on logical systems which have already been proposed in several papers. Our contributions here are (1) a new logical rule which makes for proofs which are somewhat easier to find, and (2) a soundness/completeness theorem which works for all finitary functors, in particular removing a weak pullback preservation requirement that had been used previously. Our work is based on properties of precongruence relations and also on a new parametrized coinduction principle.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases precongruence, kernel bisimulation, finitary functor, coalgebra, behavioural equivalence

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.23

1 Introduction

The goal of this paper to construct a sound and complete logic for behavioral equivalence for finitary set functors. That is, we aim to construct a logical system whose assertions include equations between variables. Given a finitary set functor F and a finite coalgebra $(X, f : X \to FX)$ we would like it to be the case that $\vdash x = y$ if and only if x and y have the same image in the final F-coalgebra. We would like the proof system to have the expected properties: for example, proofs should be finite and should be easily checkable. Moreover, even to propose a syntax which pertains to a finitary set functor F, we prefer to work with a presentation of F as a quotient of a signature functor, say H (see [5]).

We build on a line of work in this area. [15] presented a sound and complete logic for terms which denote elements of the final coalgebra of a variant of finite power set functor. This was greatly extended in [17] by moving to arbitrary finitary functors preserving weak pullbacks. Indeed the results in that paper heavily depended on preservation of weak pullbacks, as the soundness and completeness depended on properties of Aczel-Mendler bisimulations. Our first achievement is to remove the restriction to functors which preserve weak pullbacks. Working with *precongruences* (from [1]) obviates this restriction. What makes this work is the fact the final coalgebra of a finitary functor F may be described as the quotient of the final coalgebra for its related signature functor H by the greatest precongruence.

Our second achievement is a new proof rule in the logic which allows precongruences to be built up in phases. This replaces a rule in [17] which required a bisimulation to be

* This work was partially supported by a grant from the Simons Foundation (#245591 to Lawrence Moss).

"guessed all at once." The soundness of this new principle relies on properties of greatest fixed points of *parametrized precongruence relations*, a variant of precongruences, and is therefore quite different than the earlier approach in [17].

Related logical systems. The logical system in this paper is related to many others. Moschovakis's FLR_0, as presented in [11] provided logical systems for equality of recursive terms. But the semantics was presented either in terms of least fixed points on complete partial orders, or more generally in settings with a categorical fixed point operator (essentially iteration theories). [14] showed that the interpretation into final coalgebras was complete for these logics, but the only functors considered were signature functors on Set. The main work in that paper concerned constructing the semantics and checking the soundness of the rules, because FLR_0 allows terms like "x `where` $x = x$". [15] went beyond signature functors by considering a variant the finite power set functor. The syntax had term formers for sets: if t_1, …, t_n are terms, then $\{t_1, \ldots, t_n\}$ is a term. The logical rules then incorporated a version of the Axiom of Extensionality from set theory, and it was not clear how to extend this to a wider class of functors. This was done for finitary set functors in [17], using the representation of such functors as quotients of signature functors. Indeed the kernel equivalences used in the representation were directly incorporated into the inference rules. But as mentioned above, the properties of the system depended on the preservation of weak pullbacks. As in [17], the main work is in getting a usable syntax and in proving the soundness of the logic; proving the completeness is (somewhat surprisingly) an easier task.

The expression calculus of Bonsangue et al. in [8] and [7] or Milius' related work in the setting of vector spaces [13] are further related work. What we do in this paper goes in a different direction.

Outline. In Section 2, we recall basic definitions of coalgebras and kernel bisimulations as well as facts on finitary set functors and on coinduction principles. In Section 3, we recall the notion of a precongruence relation. We develop this notion from scratch, recalling results from [1, 9] that we shall use. We then specialize this notion to finitary functors and prove that the final coalgebra of a finitary functor is the quotient of the final coalgebra for its related signature functor by the greatest precongruence. In Section 4, we propose our logic for behavioral equivalence for coalgebras of finitary set functors. We use our results on precongruences to prove the soundness and completeness and completeness of our logic.

2 Preliminaries

In this section, we recall definitions and basic results about coalgebras, kernel bisimulations, and finitary functors. Our setting is the category Set, and all functors are assumed to be Set-endofunctors. Section 2.3 presents some facts on another background topic, greatest fixed points of operators.

2.1 Coalgebras

Given a Set-endofunctor F, an *F-coalgebra* is a set X together with a map $f : X \to FX$. The set is often called the *carrier* of the coalgebra, while f gives its *structure*.

An *F-coalgebra morphism* from an F-coalgebra (X, f) to another F-coalgebra (Y, g) is a map $\varphi : X \to Y$ such that the following diagram commutes:

$$\begin{array}{ccc} X & \xrightarrow{f} & FX \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle F\varphi} \\ Y & \xrightarrow{g} & FY \end{array}$$

F-coalgebras together with coalgebra morphisms between them form a category. When this category has a final object, we call it the *final F-coalgebra* and denote it by νF. Points in coalgebras which have the same image in νF are *behaviorally equivalent* to one another.

A *kernel bisimulation* from one coalgebra (X, f) to another (Y, g) is a pullback of a cospan $(X, f) \to (Z, h) \leftarrow (Y, g)$. Here we are following the name from [18], but in various other sources these are called cocongruences. In case $(X, f) = (Y, g)$, such a pullback is called a *behavioral equivalence*, even if the final F-coalgebra does not exist.

(Aczel-Mendler) bisimulations are a related notion. An (F-)*bisimulation* is a relation $R \subseteq X \times Y$ such that there is an F-coalgebra structure on R, say $\rho : R \to FR$, such that the two projections $\pi_1 : R \to X$, $\pi_2 : R \to Y$, are coalgebra morphisms.

These coalgebraic notions are parametrized by the functor F, which is often clear from context. As usual, when this is the case, we drop the prefixed F and simply say "coalgebra", "coalgebra morphism", "bisimulation", and so on.

As shown in Staton [18], Aczel-Mendler bisimulations and kernel bisimulations agree when F preserves weak pullbacks. In Chapter 1 of Kurz's thesis [12], it is shown that much of universal coalgebra [16] can be recovered when F does not preserve weak pullbacks by replacing Aczel-Mendler bisimulations with behavioral equivalences.

2.2 Finitary functors

If $\mathcal{A}$ is a category with filtered colimits, then a functor $F : \mathcal{A} \to \mathcal{B}$ is *finitary* if F preserves filtered colimits. In this paper, the main category of interest is Set. For $F : \mathsf{Set} \to \mathsf{Set}$, there are several important equivalent formulations to the notion of a finitary functor.

First: for each X and each $x \in FX$, there is a finite $Y \subseteq X$ such that $x \in Fi[FY]$, where $i : Y \hookrightarrow X$ is the inclusion. This equivalence was proven by Adámek and Porst [5]. This formulation implies that many functors of interest are finitary including the finite power set functor, $F(X) = A \times X$ for all sets A, any signature functor (with finite arities), and the discrete distribution functor.

Even more useful for our purposes is a formulation given in terms of *presentations via signatures and relations*. We begin with a signature Σ, and thus with a functor $H_\Sigma : \mathsf{Set} \to \mathsf{Set}$. For each set X, $H_\Sigma(X)$ is the set of all flat terms $f(x_1, \ldots, x_k)$, where f is a k-ary symbol in Σ, and $x_1, \ldots, x_k \in X$. For functions $\phi : X \to Y$, the functor performs a simple variable renaming: $H_\Sigma\phi : f(x_1, \ldots, x_n) \mapsto f(\phi(x_1), \ldots, \phi(x_n))$. The adjective *flat* or *basic* here refers to the fact that terms consist only of symbols applied to variables: f applied to n other non-variable terms is not a term in our context.

A *flat Σ-equation* is a pair (ℓ, r) of elements of $H_\Sigma(X)$. An *instance* of a Σ-equation is an image of each side of an equation under a renaming: $(H_\Sigma\phi(\ell), H_\Sigma\phi(r))$. Consider a fixed set of Σ-equations E with variables from S. (In our examples S is usually a finite set, though we allow it to be infinite.) The instances of E in $H_\Sigma(X)$ is the following:

$$\equiv_X \triangleq \{(H_\Sigma\phi(\ell), H_\Sigma\phi(r)) : \phi : S \to X, (\ell, r) \in E\}$$

For any X, the instances of E in $H_\Sigma(X)$ is an equivalence relation on $H_\Sigma(X)$. The assignment $X \to H_\Sigma/\equiv_X$ extends to a functor which we call H_Σ/E. Moreover, there is a natural transformation $\epsilon : H_\Sigma \to H_\Sigma/E$. All components of ϵ are surjective. We say that H_Σ/E is *presented by a finitary signature and a set of basic equations*.

▶ **Example 1.** Let $\mathcal{P}_3(X)$ be the set of subsets of X of size < 3. $\mathcal{P}_3$ is a functor in the obvious way, as a subfunctor of the covariant power set. Here is a presentation of $\mathcal{P}_3$: consider a signature with two symbols: n and $*$, with arities 0 and 2 respectively. (We write $*$ with infix notation.) Thus, $H_\Sigma(X)$ consists of n and all terms $x * y$ for $x, y \in X$. We take $S = \{s, t\}$, and just one basic equation: $s * t = t * s$. For all X, the equivalence relation $\equiv_X$ identifies $x * y$ with $y * x$. The set of equivalence classes $H_\Sigma(X)/\equiv_X$ corresponds with $\mathcal{P}_3(X)$: $[n]$ corresponds with $\emptyset$, and $[x * y]$ with $\{x, y\}$. The correspondence shows that $\mathcal{P}_3$ is naturally isomorphic to H_Σ/E.

▶ **Proposition 2** ([2], 3.12). *A functor $F : \mathsf{Set} \to \mathsf{Set}$ is finitary if and only if F is naturally isomorphic to a functor presented by a finitary signature and a set of basic equations.*

2.3 A coinduction principle for monotone operators

Our final preliminary section deals with a coinduction principle that we shall use in the soundness proof in Section 4.2. We present the work in a fairly general setting, because it will probably be easier to read and adapt this way.

Let L be a complete meet-semilattice. That is, L is a poset with greatest lower bounds $\bigwedge S$ of all sets S. Taking $S = \emptyset$ shows that L has a top element, 1. We also are interested in monotone operators $\Phi : L \to L$. By the Knaster-Tarski Theorem, Φ has a greatest post-fixed point which we write as Φ^*. To obtain Φ, we define iterates Φ^α by transfinite recursion on the ordinal α: $\Phi^0 = 1$, $\Phi^{\alpha+1} = \Phi(\Phi^\alpha)$, and for limit λ, $\Phi^\lambda = \bigwedge_{\alpha<\lambda} \Phi^\alpha$. Using the fact that L is a set, there is an ordinal α such that $\Phi^{\alpha+1} = \Phi^\alpha$. We write Φ^α as Φ^*. Φ^* is a fixed point of Φ. Moreover, if $p \leq \Phi(p)$, then $p \leq \Phi^*$.

We are frequently interested in principles which allow us to establish relationships between greatest fixed points of related operators. For any $r \in L$, we define a new operator $\Phi_r : L \to L$ by $\Phi_r(x) = \Phi(x) \vee r$. Each Φ_r is monotone, and so has a greatest fixed point, which we denote Φ_r^* as an abbreviation of $(\Phi_r)^*$. We point out $r \leq \Phi_r^*$ for all r.

▶ **Lemma 3** (Parametrized Coinduction Principle). *If $s \leq \Phi(\Phi^*_{r\vee s})$, then $\Phi^*_{r\vee s} \leq \Phi^*_r$.*

Proof. We calculate: $\Phi^*_{r\vee s} = \Phi(\Phi^*_{r\vee s}) \vee (r \vee s) = \Phi(\Phi^*_{r\vee s}) \vee r = \Phi_r(\Phi^*_{r\vee s})$. Since Φ^*_r is the greatest fixed point of Φ_r, we know $\Phi^*_{r\vee s} \leq \Phi^*_r$. ◀

We point out that $\Phi^*_r \leq \Phi^*_s$ for $r \leq s$ by montonicity arguments. One could therefore read this lemma as establishing $\Phi^*_{r\vee s} = \Phi^*_r$, provided s satisfies the condition given.

There is an obvious connection to be drawn to the work of Hur et. al. [10]. In this work, there is also a constellation of related greatest fixed points, each determined by an element in the lattice. They define $G_\Phi(r)$ to be the greatest fixed point of $\Phi'_r(x) \triangleq \Phi(x \vee r)$ and work with these $G_\Phi(r)$. Our Φ_r and their Φ'_r differ very slightly, and this subtle difference leads to different properties. For example, we point out again $r \leq \Phi^*_r$ for all r, but $r \not\leq G_\Phi(r)$. Our parametrized coinduction principle above is our analog to their accumulation theorem.

3 Precongruence relations

In this section, we recall the notion of a precongruence relation from [1]. We mention their basic properties and their connections to congruences. Then we begin specializing these results towards the context of finitary functors. The theme of this section is that the greatest precongruence relation on a coalgebra X is a greatest fixed point of a particular operator on the set of all relations on X. It turns out to be fruitful to study this operator, and we have set the stage for this in Section 2.3 just above. We also analyze the greatest fixed points of a

family of related operators. These turn out to be useful in the logic presented in the final section.

▶ Notation. If R is a relation on X, we denote its equivalence closure by $e(R)$. Given any relation R the canonical function $x \mapsto [x]_{e(R)}$ sending an element of X to its equivalence class in $X/e(R)$ is denoted q_R.

▶ **Definition 4.** Let $F : \mathsf{Set} \to \mathsf{Set}$ be a Set endofunctor and $(X, f : X \to FX)$ be an F-coalgebra. A *F-precongruence* (relation) on this coalgebra is a relation R on X such that $Fq_R(f(x)) = Fq_R(f(y))$ for all $(x, y) \in R$. In other words, $R \subseteq ker(Fq_R \circ f)$.

A *precongruence equivalence* is a precongruence relation which is also an equivalence relation.

▶ **Example 5.** We use $\mathcal{P}_3$ from Example 1. Let $X = \{x, y\}$, and consider the coalgebra structure d given by $d(x) = \{x\}$ and $d(y) = \{x, y\}$. We quickly verify $R = \{(x, y)\}$ is a precongruence. First note $[x]_{e(R)} = \{x, y\} = [y]_{e(R)}$. Then it is clear that

$$\mathcal{P}_3 q_R(d(x)) = \{[x]\} = \{[x], [y]\} = \mathcal{P}_3 q_R(d(y))$$

Aczel and Mendler [1] also introduced the related notion of a *congruence*.

▶ **Definition 6.** Given a coalgebra $(X, d : X \to FX)$, a *congruence* on this coalgebra is an equivalence relation R on X such that X/R carries a coalgebra structure $b : X/R \to F(X/R)$ and the quotient map $q_R : X \to X/R$ is a coalgebra morphism.

3.1 Basic properties of precongruence relations

In this section, we recall important properties of precongruences. We intend to leave the impression that precongruences and precongruence equivalences have much in common with bisimulations and bisimulation equivalences. However, an important difference is that the characterization of precongruence equivalence as kernels of coalgebra morphisms does **not** have any dependency on properties of the functor in question, unlike bisimulation equivalences. This is a key ingredient in being able to remove the weak pullback requirement from the scope of the logic presented in [17].

▶ **Proposition 7.** *Suppose (X, d) is an F-coalgebra. The following hold:*

1. *The diagonal Δ_X is a precongruence equivalence.*
2. *Arbitrary unions of precongruence relations are precongruence relations.*
3. *If a relation is a precongruence, so is its inverse.*
4. *If two relations containing the diagonal are precongruences, so is their composition.*
5. *A relation is a precongruence if and only if its equivalence closure is a precongruence.*

Proof. 1 and 3 follow from the definition of a precongruence.

2. Suppose R_i is a collection of precongruence relations. Let us write S for the union $\bigcup R_i$. Suppose $(x, y) \in S$. Then $(x, y) \in R_k$ for some k, and since $e(R_k) \subseteq e(S)$, there is a morphism q so that $q_S = q \circ q_{R_k}$. Since R_k is a precongruence, we know $Fq_{R_k}(d(x)) = Fq_{R_k}(d(y))$. By applying Fq to both sides, we find that S is a precongruence.

4. If R and S contain the diagonal, then their composition $R \cdot S$ contains both R and S. Suppose $(x, z) \in R \cdot S$, so there exists $y \in X$ such that $(x, y) \in R$ and $(y, z) \in S$. Since R and S are precongruences, we know that $Fq_R(d(x)) = Fq_R(d(y))$ and $Fq_S(d(y)) = Fq_S(d(z))$. This implies $Fq_{R \cdot S}(d(x)) = Fq_{R \cdot S}(d(y))$ and $Fq_{R \cdot S}(d(y)) = Fq_{R \cdot S}(d(z))$, as desired.

5. The above properties can be combined to show that if a relation is a precongruence, its equivalence closure is as well. If the equivalence closure of a relation is a precongruence, a fortiori that relation is a precongruence as well. ◀

Proposition 7 implies the following result, fundamental to the rest of our work.

▶ **Theorem 8** ([1]). *For every F-coalgebra (X, f), there is a largest precongruence R on X. R is an equivalence relation.*

Proof. Here is a sketch; we shall see a different proof of it in Section 3.4. Precongruence relations being closed under unions also implies that there is a greatest precongruence on any given coalgebra. This greatest precongruence must also be an equivalence relation by the fifth property above. ◀

▶ Remark. In contrast to congruences, precongruences are not closed under intersections. To see this, define an Id-coalgebra on $X = \{x, y, z\}$ by $d(x) = y$, $d(y) = z$ and $d(z) = x$. Then the two relations $R = \{(x, y), (y, z)\}$ and $S = \{(x, y), (z, y)\}$ are both precongruences, but their intersection is not.

Next we recall results about precongruence equivalences. A particularly important result for our purposes is that they coincide with kernels of coalgebra morphisms.

▶ **Theorem 9** ([1], Lemma 5.1). *Let (X, d) be an F-coalgebra and let $R \subseteq X \times X$. The following are equivalent:*
1. *R is a congruence on X.*
2. *R is the kernel of a coalgebra morphism φ with domain (X, d).*
3. *R is a precongruence equivalence on X.*

We can use these results to establish a relationship between precongruences and bisimulations:

▶ **Corollary 10.** *Suppose that (X, d) is an F-coalgebra. Then R is a precongruence relation on X if and only if $e(R)$ is a congruence. If F moreover preserves weak pullbacks, then R is a precongruence relation on X if and only if $e(R)$ is a bisimulation.*

Proof. We know R is a precongruence if and only if $e(R)$ is a precongruence. By Theorem 9, $e(R)$ is a precongruence if and only if it is the kernel of an F-coalgebra morphism. When F preserves weak pullbacks, kernels of coalgebra morphisms exactly coincide with bisimulation equivalences, see [16, Proposition 5.7, 5.8]. ◀

In particular, the behavioral equivalence relation has to be a precongruence equivalence.

▶ **Proposition 11.** *Let F be a Set-endofunctor and suppose the final coalgebra of F, νF, exists. If (X, d) an F-coalgebra, then the kernel of the final coalgebra map $h_X : X \to \nu F$ is the greatest precongruence relation on X.*

Proof. Let R be the greatest precongruence relation on X. By Theorem 9, $ker(h_X)$ is a precongruence equivalence and hence $ker(h_X) \subseteq R$.

Also by Theorem 9, we know $q_R : X \to X/R$ is an F-coalgebra morphism. Since X/R is an F-coalgebra it also has a final morphism into νF, which we denote $h_{X/R}$. By finality, $h_X = h_{X/R} \circ q_R$. Therefore, $R = ker(q_R) \subseteq ker(h_X)$. ◀

3.2 Precongruence relations across natural transformations

Suppose $\epsilon : H \to F$ is a natural transformation. Then each H-coalgebra (X, d) carries an F-coalgebra structure $\epsilon_X \circ d$. The relationship between H and F leads to relationships between other coalgebraic notions, including precongruence relations, on H- and F-coalgebras.

▶ **Definition 12.** Let $\epsilon : H \to F$ be a natural transformation between Set endofunctors. We define a collection of equivalence relations $=_{\epsilon,X}$ as $ker(\epsilon_X)$. Typically the relevant set X can be inferred from context, and we write $=_\epsilon$ instead.

If we assume further ϵ is epic, there is a very close relationship between the final H- and final F-coalgebras. This assumption holds in the context of finitary functors; recall Section 2.2.

▶ **Lemma 13** ([9], Lemma 2.3)**.** *Suppose $\epsilon : H \to F$ is an epic natural transformation and that the final H-coalgebra, $(\nu H, t)$, exists. Then νF exists and is the quotient of the F-coalgebra $(\nu H, \epsilon_{\nu H} \circ t)$ by the greatest F-precongruence relation R on it.*

▶ **Theorem 14** (Coinduction Principle)**.** *In the setting of Lemma 13, consider νH as an F-coalgebra with structure $\epsilon_{\nu H} \circ t$. Let $\varphi : \nu H \to \nu F$ be the final F-coalgebra map. Let R be any relation on νH with the property that if $(u, v) \in R$, then $(Hq_R \circ t)(u) = (Hq_R \circ t)(v)$. Then for all pairs $(u, v) \in R$, $\varphi(u) = \varphi(v)$.*

Proof. We claim that R is an F-precongruence on $(\nu H, \epsilon_{\nu H} \circ t)$. To see this, let $(u, v) \in R$. By naturality, $Fq_R \circ \epsilon_X \circ t = \epsilon_{X/R} \circ Hq_R \circ t$. So u and v have the same image under $(Fq_R \circ \epsilon_X) \circ t$. This proves our claim. By Lemma 13, $R \subseteq ker(\varphi)$. ◀

3.3 The final coalgebra for finitary functors: level cuts and precongruences

Recall that a finitary set functor F may be described in terms of a signature functor H and a natural transformation $\epsilon : H \to F$ with surjective components. Lemma 13 gives the relation between νF and νH: νF is the quotient of νH by the greatest precongruence relation on νH. The idea in this section is that since we have a convenient representation of νH as the set of all (finite and infinite trees) on H, we have a tool to study νF. The main point of this section is to mention two ways to carry out this study.

Adámek and Milius characterized νF in [3] as the quotient of νH by a relation $\sim^*$ based on level cuts. We briefly describe this relation and compare it to our characterization with an example.

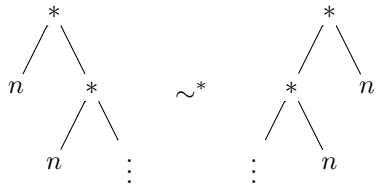
Let $\sim$ be the least congruence on finite trees which contains the relation $=_{\epsilon,X\cup\{\bot\}}$. The *level k cutting* of a tree $\sigma \in \nu H$ is defined to be the biggest height k subtree of σ with the level k leaves replaced by a fresh symbol $\bot$. We define the relation $\sim^*$ on νH by $\sigma \sim^* \tau$ if and only if the level k cuts of σ and τ are related by $\sim$ for all $k \in \omega$.

▶ **Theorem 15** (Adámek and Milius, [3])**.** *$(\nu H)/\sim^*$ is the final F-coalgebra.*

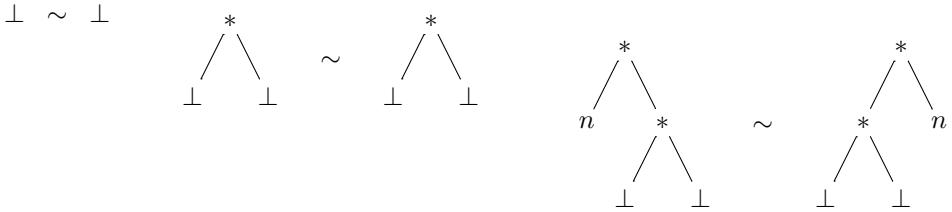
We compare these two characterizations with an example. It is based on the presentation in Example 1.

▶ **Example 16.** Let $X = \{x, y, z\}$, and let d be the H-coalgebra $d(x) = z * x$, $d(y) = y * z$ and $d(z) = n$. Then d extends to a $\mathcal{P}_3$-coalgebra via ϵ_X in the usual way. We claim that x and y have the same image in the final $\mathcal{P}_3$-coalgebra.

The tree unfolding of x and y in νH, the final H-coalgebra, look like the following:



Intuitively, these two trees are the same as $\mathcal{P}_3$-coalgebras since ϵ makes the $*$ operation commutative. To show this with level cuts, we have to show that the level k cuts of the above trees are pairwise related by $\sim$ for all $k \in \omega$. This sequence starts:



This is easy enough to do by induction, but it does require an induction.

We mentioned that the purpose of this section is to present two ways of studying νF using νH. The second way is to use Theorem 14.

▶ **Example 17.** We revisit Example 16 from just above. Again, we wish to see that x and y have the same images in the final F-coalgebra. With precongruence relations, we need only exhibit a relation such that all pairs $(u, v) \in R$ satisfy $Hq_R(d(u)) =_\epsilon Hq_R(d(v))$. We suggest $R = \{(x, y)\}$. Here are the calculations that verify R is a precongruence:

$$Hq_R(d(x)) = Hq_R(z * x) = [z]_{e(R)} * [x]_{e(R)}$$
$$Hq_R(d(y)) = Hq_R(y * z) = [y]_{e(R)} * [z]_{e(R)} = [x]_{e(R)} * [z]_{e(R)}$$

and in fact $[z]_{e(R)} * [x]_{e(R)} =_\epsilon [x]_{e(R)} * [z]_{e(R)}$. (The reason comes from Example 1: the operation $*$ is interpreted by a commutative function on every quotient set $H(X)/\equiv_X$.) So R is a precongruence. By Theorem 14, x and y have the same image in the final $\mathcal{P}_3$-coalgebra.

There is no induction in Example 17, and the argument has a flavor closer to a coinductive proof. That is, we start the proof assuming that $(x, y) \in R$, and we use this very fact when we claim that $[x]_{e(R)} = [y]_{e(R)}$. Of course, we hasten to add that by a "coinductive" (or "circular") proof we mean a proof that is justified by a sound principle. In our setting, this sound principle is Proposition 11.

The rest of this paper elaborates the observation in Example 17 into a proof system and studies it in connection with the coinduction principles which we have seen already.

3.4 Precongruence iterates and parametrized precongruence relations

For any set X, the set of all relations on X is a complete lattice. Its top element is $X \times X$. Given an F-coalgebra (X, f), define the following operation on this lattice:

$$\begin{aligned} R^- &= \{(x, y) \in X \times X : Fq_R(f(x)) = Fq_R(f(y))\} \\ &= ker(Fq_R \circ f) \end{aligned}$$

Note that the post-fixed points of $R \mapsto R^-$ are precisely the precongruence relations. This is our motivation in investigating this operation.

▶ **Proposition 18.** *If $R \subseteq S$, then $R^- \subseteq S^-$.*

Proof. Suppose that $R \subseteq S$, and let $(x, y) \in R^-$. This means $Fq_R(f(x)) = Fq_R(f(y))$. From $e(R) \subseteq e(S)$ and we infer a morphism q such that $q_S = q \circ q_R$. Applying Fq to both sides yields $(x, y) \in S^-$. ◀

Having a monotone operator on a complete lattice, we can instantiate the results of Section 2.3 in this context. Following that approach, we define additional monotone operators on $\mathcal{P}(X \times X)$: for each relation S,

$$R^-_S = R^- \cup S = \{(x, y) : Fq_R(f(x)) = Fq_R(f(y))\} \cup S.$$

Conflating S and $(\cdot)^-_S$ slightly, we have *iterates* S^α for ordinals α as defined in Section 2.3.

$$S^0 = X \times X \quad \text{and} \quad S^{\alpha+1} = (S^\alpha)^-_S \quad \text{and} \quad S^\lambda = \bigcap_{\alpha<\lambda} S^\alpha$$

We define S^* to be the greatest fixed point of $(\cdot)^-_S$.

When F is a finitary set functor presented by a signature functor H and an epic natural transformation ϵ, the F-coalgebra (X, f) has a related H-coalgebra, which we denote (X, d), satisfying $\epsilon_X \circ d = f$.

Then we can alternatively define $R^- \triangleq ker(Fq_R \circ f)$ using $Fq_R \circ \epsilon_X = \epsilon_{X/R} \circ Hq_R$ by the naturality of ϵ:

$$R^- = \{(x, y) : Hq_R(d(x)) =_\epsilon Hq_R(d(y))\}. \tag{1}$$

This is a more useful formulation in what follows.

▶ **Proposition 19.** *Let R and S be any relations on X. If $S \subseteq ((R \cup S)^*)^-$, then $(R \cup S)^* \subseteq R^*$.*

Proof. This is just a restatement of Lemma 3 in our current context. ◀

To illustrate the value of the iterative approach, we present a different proof of Aczel and Mendler's Theorem 8. The operator $(\cdot)^-$ is a monotone operator on $\mathcal{P}(X \times X)$, and hence some iterate $\emptyset^\alpha$ is a greatest post-fixed point. This relation $\emptyset^\alpha$ is a precongruence, and indeed it includes all precongruences. An easy induction shows that each iterate $\emptyset^\beta$ is an equivalence relation, using the facts that each R^- is the kernel relation of some morphism, and also that the intersection of any family of equivalence relations is an equivalence relation.

▶ Remark. We think of the iteration of $(\cdot)^-_S$ as a partition refinement process. We start with the complete relation on X, and at each step we retain only the pairs which satisfy $Hq_R(d(x)) =_\epsilon Hq_R(d(y))$. The iteration converges in ω steps for a finitary functor.

The relation $\emptyset^*$ is the greatest precongruence relation. The variant operations $(\cdot)^-_S$ also have greatest fixed points; these are not necessarily precongruences. They include this greatest precongruence relation, but in a somewhat controlled way. We call these *parametrized precongruences*. As we shall see in Example 20 below, parametrized precongruences need not be equivalence relations.

The relation S in the $(\cdot)^-_S$ operation provides a "forgiveness parameter": all $(x, y) \in S$ remain in all stages of the S^i iteration even if they do not satisfy the refinement predicate $Hq_R(d(x)) =_\epsilon Hq_R(d(y))$. Since pairs in S stubbornly remain in each iterate S^i, other pairs which separate in $\emptyset^j$ to fail to separate in S^j, even if those pairs themselves are not in S! We illustrate such a scenario in the next example.

$$\frac{}{R \vdash \sigma = \sigma}\ r \qquad \frac{R \vdash \tau = \sigma}{R \vdash \sigma = \tau}\ s \qquad \frac{R \vdash \sigma = \tau \quad R \vdash \tau = \rho}{R \vdash \sigma = \rho}\ t$$

$$\frac{}{\{(x,y)\} \cup R \vdash x = y}\ a \qquad \frac{R \vdash x_1 = y_1 \quad \dots \quad R \vdash x_{ar(f)} = y_{ar(f)}}{R \vdash f(x_1, \dots, x_{ar(f)}) = f(y_1, \dots, y_{ar(f)})}\ c$$

$$\frac{\alpha =_\epsilon \beta}{R \vdash \alpha = \beta}\ \epsilon \qquad \frac{R \cup S \vdash \sigma = \tau \quad \forall (x,y) \in S. R \cup S \vdash d(x) = d(y)}{R \vdash \sigma = \tau}\ i$$

Figure 1 Our proof system.

▶ **Example 20.** Recall the signature for the functor $\mathcal{P}_3$ from Example 1. Define a $\mathcal{P}_3$-coalgebra structure on $X = \{x, y, z\}$ by extending the following signature coalgebra with ϵ_X.

$$d(x) = z * y \qquad d(y) = z * z \qquad d(z) = n$$

We consider the $\emptyset^*$ iteration for this coalgebra. We know $\emptyset^1 = \Delta_X \cup \{(x,y),(y,x)\}$ since $[-]_{\emptyset^0} * [-]_{\emptyset^0} \neq_\epsilon [n]$ for any variables in the left hand side, so x and y are separated from z. Then $\emptyset^2 = \Delta_X$ since $[z]_{\emptyset^1} * [y]_{\emptyset^1} \neq_\epsilon [z]_{\emptyset^1} * [z]_{\emptyset^1}$. After this the sequence stabilizes at Δ_X. We shall see that this implies that all three of these variables have different images in the final coalgebra.

Next we take $S = \{(y,z)\}$ and consider the S^* iterates for the same coalgebra. We know $S^1 = \emptyset^1 \cup S = \Delta_X \cup \{(x,y),(y,x)\} \cup \{(y,z)\}$. Notice the equivalence closure of S^1 is still the total relation on X. As a consequence, $S^2 = S^1$, so we have stabilized after only one step.

We certainly would have expected $(y,z) \in S$ since that pair is returned to the relation during each application of $(\cdot)^-_S$. However, (x,y) is also present in S^* despite not being in S. Intuitively, this is because their separation in the $\emptyset^*$ iteration required y and z to separate in the $\emptyset^1$ iterate. This never happens in the S^i process.

4 Logic

Let F be finitary, and assume that it is presented via $\epsilon : H = H_\Sigma \to F$ which has a set E of Σ-equations, as in Section 2.2.

Let X be any set. We again think of the elements of X as variables, but they could be any objects. For each H-coalgebra (X,d), we have a proof system that is intended to talk about the final F-coalgebra map of $\epsilon_X \circ d$.

In the syntax of our logic, $x, y \in X$ are variables, $\alpha, \beta \in HX$ are flat terms, and σ, τ are either both variables or both flat terms. We do not have terms that are "deeper" than flat terms. The judgments are of the form $R \vdash \sigma = \tau$ where $R \subseteq X \times X$ is a relation on variables. The role of R on the left side of the turnstile is to track the precongruence relation we are verifying through the course of the proof.

4.1 Proof system

For each H-coalgebra (X,d), Figure 1 contains a proof system for determining behavioral equivalence in νF. The function d enters into the i rule. (That is, the proof system depends on the underlying coalgebra.)

The first three rules, r, s, and t, are the obvious equivalence properties, and c just says that each function symbol is a congruence. In ϵ, we mean to take as axioms all of the instances of the equations of E in the carrier set X of the coalgebra at hand.

The way to think about the i rule as a form of *strong coinduction*. Just as proof by induction frequently appears in strong forms, the same is true of coinduction. The reader will want to look at examples, since this rule is the main innovation in our system. It is a proof-theoretic version of what we saw in Lemma 3.

We write $R \vdash \sigma = \tau$ to mean that there is a tree of finite height whose leaves are blank, whose root is $R \vdash \sigma = \tau$, and all of whose internal nodes are justified by one of the rules in the system. We point out that instances of the i rule with infinite S will create "infinitely wide" proofs. These are allowable in our system; we only require finite height.

▶ **Example 21.** Here is the formal version of the argument presented in Example 17. Consider the coalgebra $d : X \to H_\Sigma X$, where $X = \{x, y, z\}$ and where

$$d(x) = x * z \qquad d(y) = z * y \qquad d(z) = n$$

Let $S = \{(x, y)\}$. The proof tree below witnesses $\vdash x = y$.

$$\dfrac{\dfrac{}{S \vdash x = y}\, a \qquad \dfrac{\dfrac{}{S \vdash x * z = z * x}\, \epsilon \qquad \dfrac{\dfrac{}{S \vdash z = z}\, r \quad \dfrac{}{S \vdash x = y}\, a}{S \vdash z * x = z * y}\, c}{S \vdash x * z = z * y}\, t}{\vdash x = y}\, i$$

The i rule here is applied with $R = \emptyset$ and S as above.

▶ Remark. This proof system is very similar to the one developed in [17], but the i rule is slightly different. The system in that paper instead had the rule b

$$\dfrac{S \vdash \sigma = \tau \quad \forall (x, y) \in S . S \vdash d(x) = d(y)}{\vdash \sigma = \tau}\, b$$

b is derivable from i: take $R = \emptyset$ in i. Using the weaker rule i, we could obtain the completeness of the system with respect to semantics which we discuss below. However, the i rule is not derivable from b, and is much easier to use. Rather than having to verify and discharge a precongruence relation all as one step, the i rule allows the precongruence relation to be built up in multiple steps.

4.2 Soundness and completeness

Recall that we start with an H-coalgebra (X, d), and we are interested in the images of points $x, y \in X$ in the final coalgebra of F (not H). As Example 21 suggests, an assertion $\vdash x = y$ should mean that those images coincide. We formally define our semantics as behavioral equivalence: $\vDash x = y$ for two variables in X means x and y have the same image in νF. But to prove soundness by induction on proof trees means proving something about inferences with hypotheses (the R on the left), and this is our next result.

▶ **Proposition 22** (Strong soundness). *Suppose (X, d) is an H-coalgebra, $x, y \in X$ and $\alpha, \beta \in HX$. Then $R \vdash x = y$ implies $(x, y) \in e(R^*) = ker(q_{R^*})$, and $R \vdash \alpha = \beta$ implies $Hq_{R^*}(\alpha) =_\epsilon Hq_{R^*}(\beta)$. (Equivalently, $(\alpha, \beta) \in ker(\epsilon_{X/e(R^*)} \circ Hq_{R^*})$.)*

Proof. By induction on the height of proof trees in our proof system. Both relations mentioned in the consequents above are equivalence relations; the r, s and t rules follow. The a rule follows from $R \subseteq e(R^*)$.

To verify the soundness of the c rule, fix an instance of the rule, say with a function symbol f in Σ. Let $n = ar(f)$. Let us write q for q_{R^*} to save on notation in this paragraph. By induction hypothesis, for $i \leq n$, $q(x_i) = q(y_i)$. And so

$$Hq(f(x_1, \ldots, x_n)) = f(q(x_1), \ldots, q(x_n)) = f(q(y_1), \ldots, q(y_n)) = Hq(f(y_1, \ldots, y_n)).$$

For the ϵ rule, asume that $\alpha =_\epsilon \beta$. Then $Hq_\emptyset(\alpha) =_\epsilon Hq_\emptyset(\beta)$. So $Hq_{R^*}(\alpha) =_\epsilon Hq_{R^*}(\beta)$.

The most interesting point is the soundness of the i rule. Fix an instance of this rule and assume our soundness result for all premises of our instance. In particular, from the rightmost premises in the i rule, we know $Hq_{(R\cup S)^*}(d(x)) =_\epsilon Hq_{(R\cup S)^*}(d(y))$ for all $(x, y) \in S$. Thus $S \subseteq ((R \cup S)^*)^-$. By Proposition 19, $(R \cup S)^* \subseteq R^*$.

If $\sigma = \tau$ is an equality of variables, say $u = v$, then the induction hypothesis for the left premise of the i rule is $q_{(R\cup S)^*}(u) = q_{(R\cup S)^*}(v)$. We just showed $(R \cup S)^* \subseteq R^*$, and thus we have $q_{R^*}(u) = q_{R^*}(v)$, as desired.

On the other hand, if $\sigma = \tau$ is an equality of flat terms, then the left induction hypothesis is $Hq_{(R\cup S)^*}(\alpha) =_\epsilon Hq_{(R\cup S)^*}(\beta)$. Again since $(R \cup S)^* \subseteq R^*$ we get our desired result of $Hq_{R^*}(\alpha) =_\epsilon Hq_{R^*}(\beta)$. ◀

▶ **Corollary 23** (Soundness). *If $\vdash x = y$, then $\vDash x = y$.*

Proof. Assume that $\vdash x = y$. By Proposition 22, $(x, y) \in ker(q_{\emptyset^*})$. But $\emptyset^*$ is a precongruence relation (indeed the largest such), and so we are done by Theorem 14. ◀

As usual for logics pertaining to (co-)recursively-defined terms, completeness is easier than soundness.

▶ **Proposition 24.** *Suppose that R is a relation on X. Then the following hold:*
1. *If $(x, y) \in e(R)$, then $R \vdash x = y$.*
2. *If $Hq_R(\alpha) =_\epsilon Hq_R(\beta)$, then $R \vdash \alpha = \beta$.*
3. *If R is a precongruence on X, then $\vdash x = y$ for all $(x, y) \in R$.*

Proof. 1. Repeated use of r, s and t.

2. Let $r : X \to X$ send each variable x to a canonical representative for the $[x]_{e(R)}$ equivalence class. Then $R \vdash \alpha = Hr(\alpha)$ and $R \vdash \beta = Hr(\beta)$ using rule c and then item 1 repeatedly. We are also using the fact that all arities of symbols in the signature corresponding to H are finite. The fact that $Hq_R(\alpha) =_\epsilon Hq_R(\beta)$ implies $Hr(\alpha) =_\epsilon Hr(\beta)$. So $R \vdash Hr(\alpha) = Hr(\beta)$ by the ϵ rule. Therefore, $R \vdash \alpha = \beta$ with s and two applications of t.

3. $R \vdash x = y$ by the a rule. By R precongruence and item 2, we know $R \vdash d(z) = d(w)$ for all $(z, w) \in R$. Therefore by i (or even by b), we have $\vdash x = y$. ◀

▶ **Corollary 25** (Completeness). *If $\vDash x = y$, then $\vdash x = y$.*

Proof. By Proposition 24(3), and the fact that $\{(x, y) | \vDash x = y\}$ is a precongruence on (X, d). ◀

▶ Remark. We are mostly interested in this proof system when X is a finite set. In that case, whenever $\vDash x = y$, there is a finite proof tree showing that $\vdash x = y$. But when X is infinite, we might need a proof tree of "infinite width" (see Example 26 below).

We could generalize our proof system and results to consider *accesssible* set functors. These have a presentation in terms of signatures and sets of equations, but the arities in the signature may be infinite. The results of this paper generalize. If κ is an infinite regular cardinal and F a κ-accessible functor, we would need proof trees of height $\leq \kappa$.

▶ **Example 26.** Write $R \vdash_* \phi$ to mean that there is a *finite* proof tree showing $R \vdash \phi$. Since we are already assuming proofs are of finite height, this finiteness condition requires that additionally proofs are finitely wide. The purpose of this example is to show that our completeness result above may sometimes require these infinitely wide proofs.

Let Σ consist of a unary function symbol, f. Let H be the associated signature functor, and let $F = H$. In particular, this implies there are no ϵ-axioms in our logic for H-coalgebras. Give $X = \{x_k : k \in \omega\}$ an H-coalgebra structure by $d(x_k) = f(x_{k+1})$.

We first note $\vdash x_0 = x_1$: set $R = \{(x_k, x_{k+1}) : k \in \omega\}$ and observe that $R \vdash_* f(x_{k+1}) = f(x_{k+2})$ for all k. Then one application of rule i with infinitely many premises shows that $\vdash x_0 = x_1$.

In contrast, we also claim that $\not\vdash_* x_0 = x_1$. The exact induction argument is a bit long for this section, but the idea is that $R \vdash_* x_j = x_k$ with $j \neq k$ requires that j and k are both less than the greatest index mentioned in R. This induction makes critical use of the fact that the i rule can only be invoked with a finite sized S. Then taking $R = \emptyset$ gives our desired claim.

4.3 Further examples

As noted above, this proof system is an improvement in two ways over the system presented in [17]. First, it is sound and complete for **all** finitary functors, not just the ones which preserve weak pullbacks. Second, the i rule is more flexible than the b rule as it allows for more gradual assumption introduction. The next two example proofs showcase these capabilities.

▶ **Example 27.** The Aczel-Mendler functor $(-)_2^3$ is defined as $(X)_2^3 = \{(x, y, z) \in X^3 : |\{x, y, z\}| \leq 2\}$. This functor does not preserve weak pullbacks, but is finitary with presentation given by $\Sigma = \Sigma_2 = \{\sigma, \tau, \rho\}$. The associated natural transformation is given by

$$\epsilon_X : \sigma(x, y) \mapsto (x, x, y) \qquad \epsilon_X : \tau(x, y) \mapsto (y, x, x) \qquad \epsilon_X : \rho(x, y) \mapsto (x, y, x)$$

Therefore, the only equations for this presentation take the form $\sigma(x, x) =_\epsilon \tau(x, x) =_\epsilon \rho(x, x)$. The final coalgebra for this functor is a singleton set, so all states in all $(-)_2^3$-coalgebras are behaviorally equivalent. Therefore, we consider the functor $1 + (-)_2^3$ instead. The presentation for this functor adds a fresh constant $*$ for the left part of the coproduct. There are no additional equations.

Consider the following coalgebra on $X = \{a, b, c\}$:

$$d(a) = \sigma(a, b) \qquad d(b) = \rho(a, a) \qquad d(c) = \tau(b, a)$$

The greatest bisimulation on the related $(-)_2^3$-coalgebra is not transitive. In particular, it does not relate a and c. However, a and c are behaviorally equivalent, as can be seen below. Take $R = \{(a, c), (a, b)\}$, then:

$$\dfrac{\dfrac{}{R \vdash a = c}\, a \qquad \dfrac{\dfrac{\dfrac{}{R \vdash a = a}\, r \qquad \dfrac{R \vdash a = b}{R \vdash b = a}\, s}{R \vdash \sigma(a, b) = \sigma(a, a)}\, c \qquad \dfrac{}{R \vdash \sigma(a, a) = \rho(a, a)}\, \epsilon}{R \vdash \sigma(a, b) = \rho(a, a)}\, t \qquad \ldots \text{(proof continues below)}}{\vdash a = c}$$

$$\ldots \qquad \dfrac{\dfrac{\dfrac{}{R \vdash a = a}\, r \qquad \dfrac{R \vdash a = b}{R \vdash b = a}\, s}{R \vdash \sigma(a, b) = \sigma(a, a)}\, c \qquad \dfrac{}{R \vdash \sigma(a, a) = \tau(a, a)}\, \epsilon \qquad \dfrac{\dfrac{}{R \vdash a = b}\, a \qquad \dfrac{}{R \vdash a = a}\, r}{R \vdash \tau(a, a) = \tau(b, a)}\, c}{R \vdash \sigma(a, b) = \tau(b, a)}\, t \times 2\ i$$

Finally, we introduce a toy signature to show the flexibility of the i rule.

▶ **Example 28.** Let $\Sigma_1 = \{f, g\}$ and define an H-coalgebra on $\{x, y, w, z, u\}$ by

$$d(x) = f(w) \qquad d(y) = f(z) \qquad d(w) = g(y) \qquad d(z) = g(u) \qquad d(u) = f(w)$$

Our goal here is to show x and y have the same image in the final coalgebra, and we look forward enough to know that for this to be true we would need $f(w) = f(z)$. Being a bit impatient, we start the proof right away with a coinductive hypothesis $R = \{(x, y), (w, z)\}$.

$$\dfrac{\dfrac{}{R \vdash x = y}\, a \quad \dfrac{\dfrac{}{R \vdash y = z}\, a}{R \vdash f(y) = f(z)}\, c \quad \dfrac{\dfrac{}{R \vdash y = u}\, ??}{R \vdash g(y) = g(u)}\, c}{\vdash x = y}\, i$$

(In the application of i in this tree, our R takes the role of S in the rule statement and the R in the rule statement is empty.) At this point we get stuck—there is nothing to help prove $y = u$. Due to i though, we can add this pair to our coinduction hypothesis on the fly. Let $S = \{(y, u)\}$. We can then finish the proof as below:

$$\dfrac{\dfrac{}{R \vdash x = y}\, a \quad \dfrac{\dfrac{}{R \vdash y = z}\, a}{R \vdash f(y) = f(z)}\, c \quad \dfrac{\dfrac{\dfrac{}{R \cup S \vdash y = u}\, a \quad \dfrac{\dfrac{}{R \cup S \vdash w = z}\, a}{R \cup S \vdash z = w}\, s}{R \cup S \vdash f(z) = f(w)}\, c}{R \vdash y = u}\, i}{R \vdash g(y) = g(u)}\, c}{\vdash x = y}\, i$$

5 Conclusion and future directions

We presented a sound and complete logic for behavioral equivalence. The core principles used in our soundness and completeness proofs were basic results on precongruence relations. We used precongruences in the context of finitary Set functors to do two things. First, we described the final coalgebra for a finitary functor as the quotient of a related final coalgebra by its greatest precongruence relation. Then we realized the greatest precongruence relation as the greatest fixed point of an operator on relations, and studied the a variant of the operator, leading to parametrized precongruences and then the the soundness of the main rule in our logical system.

Though we used several features of Set throughout this work, these features do not appear to be **unique** features of Set. We believe it would be possible to obtain similar results in a wider class of categories. In particular, we are interested in applications to order categories such as preorders and posets. Complete metric spaces would be another appropriate setting. One way to start would be to use results in [4] on generalizations of presentations of finitary functors to all locally presentable categories, and [6] for the specific case of order categories.

Another direction of generalization would be to enhance the expressivity of the logic by allowing more than just flat terms, removing the restrictions on the sets of equations, or otherwise expanding the set of available formulas. In particular, allowing partial specifications where some variables in the set of equations are undefined is upcoming work in the first author's thesis.

Acknowledgements. The authors are grateful to the anonymous CALCO referees for pointing out shortcomings of a previous version of this paper.

References

1. Peter Aczel and Nax Mendler. A final coalgebra theorem. In David H. Pitt, David E. Rydeheard, Peter Dybjer, Andrew M. Pitts, and Axel Poigné, editors, *Category Theory and Computer Science: Manchester, UK, September 5–8, 1989 Proceedings*, pages 357–365. Springer Berlin Heidelberg, Berlin, Heidelberg, 1989.
2. Jirí Adámek, H Peter Gumm, and Vera Trnková. Presentation of set functors: a coalgebraic perspective. *Journal of Logic and Computation*, 20(5):991–1015, 2010.
3. Jiří Adámek and Stefan Milius. Terminal coalgebras and free iterative theories. *Information and Computation*, 204(7):1139 – 1172, 2006.
4. Jiří Adámek, Stefan Milius, and Lawrence S Moss. On finitary functors and their presentations. In *Coalgebraic Methods in Computer Science*, pages 51–70. Springer, 2012.
5. Jiří Adámek and Hans-Eberhardt Porst. On tree coalgebras and coalgebra presentations. *Theoret. Comput. Sci.*, 311:257–283, 2004.
6. Adriana Balan and Alexander Kurz. Finitary functors: From set to preord and poset. In *Algebra and Coalgebra in Computer Science*, volume 6859 of *LNCS*, pages 85–99. Springer, 2011.
7. Marcello Bonsangue, Jan Rutten, and Alexandra Silva. An algebra for Kripke polynomial coalgebras. In *Logic In Computer Science, 2009. LICS'09. 24th Annual IEEE Symposium on*, pages 49–58. IEEE, 2009.
8. Marcello Bonsangue, Jan Rutten, and Alexandra Silva. A Kleene theorem for polynomial coalgebras. In *Foundations of Software Science and Computational Structures*, pages 122–136. Springer, 2009.
9. H Peter Gumm and Tobias Schröder. Coalgebras of bounded type. *Mathematical Structures in Computer Science*, 12(05):565–578, 2002.
10. Chung-Kil Hur, Georg Neis, Derek Dreyer, and Viktor Vafeiadis. The power of parameterization in coinductive proof. *ACM SIGPLAN Notices*, 48(1):193–206, 2013.
11. Antonius J. C. Hurkens, Monica McArthur, Yiannis N. Moschovakis, Lawrence S. Moss, and Glen T. Whitney. The logic of recursive equations. *The Journal of Symbolic Logic*, 63(02):451–478, 1998.
12. Alexander Kurz. *Logics for Coalgebras and Applications to Computer Science*. PhD thesis, Ludwig-Maximilians-Universität München, 2000.
13. Stefan Milius. A sound and complete calculus for finite stream circuits. In *Logic in Computer Science (LICS), 2010 25th Annual IEEE Symposium on*, pages 421–430. IEEE, 2010.
14. Lawrence S Moss. Recursion and corecursion have the same equational logic. *Theoretical Computer Science*, 294(1):233–267, 2003.
15. Lawrence S Moss, Erik Wennstrom, and Glen T Whitney. A complete logical system for the equality of recursive terms for sets. In *Logic and Program Semantics*, pages 180–203. Springer, 2012.
16. Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249(1):3–80, 2000.
17. David Sprunger. A complete logic for behavioural equivalence in coalgebras of finitary set functors. In *International Workshop on Coalgebraic Methods in Computer Science*, pages 156–173. Springer, 2016.
18. Sam Staton. Relating coalgebraic notions of bisimulation. *Logical Methods in Computer Science*, 7(1), 2011.

Finite Behaviours and Finitary Corecursion*

Henning Urbat

Institut für Theoretische Informatik, Technische Universität Braunschweig, Germany

Abstract

In the coalgebraic approach to state-based systems, semantics is captured up to behavioural equivalence by special coalgebras such as the final coalgebra, the final locally finitely presentable coalgebra (Adámek, Milius, and Velebil), or the final locally finitely generated coalgebra (Milius, Pattinson, and Wißmann). The choice of the proper semantic domain is determined by finiteness restrictions imposed on the systems of interest. We propose a unifying perspective by introducing the concept of a final locally $(\mathbb{I}, \mathcal{M})$-presentable coalgebra, where the two parameters $\mathbb{I}$ and $\mathcal{M}$ determine what a "finite" system is. Under suitable conditions on the categories and type functors, we show that the final locally $(\mathbb{I}, \mathcal{M})$-presentable coalgebra exists and coincides with the initial $(\mathbb{I}, \mathcal{M})$-iterative algebra, thereby putting a common roof over several results on iterative, fg-iterative and completely iterative algebras that were given a separate treatment before.

1998 ACM Subject Classification F.4.0 Mathematical Logic and Formal Languages – General

Keywords and phrases Iterative algebra, completely iterative algebra, fg-iterative algebra, rational fixpoint, terminal coalgebra, iterative monad

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.24

1 Introduction

Coalgebras model a wide variety of state-based systems, including deterministic and nondeterministic automata, weighted automata, labelled transition systems and probabilistic systems. One striking advantage of the coalgebraic approach is its beautiful account of semantics: the possible behaviours of states are captured by the final coalgebra, and the behaviour of a system is given by its final homomorphism. As a prominent example, the set functor $\{0,1\} \times \mathsf{Id}^\Sigma$ (modelling deterministic automata) has a final coalgebra carried by the set of all languages over Σ, and the final homomorphism maps a state of an automaton to its accepted language [21].

In computer science, the focus is typically on systems satisfying some finiteness restrictions. For example, classical automata theory investigates finite automata (i.e. automata with finitely many states and input symbols) and their behaviours, the regular languages. And in the recently developed theory of nominal automata [9], the objects of interest are automata with possibly infinite, but orbit-finite nominal sets of states and inputs. From a "finite" point of view, the semantics given by the final coalgebra is sometimes unsatisfactory because it may identify states of finite coalgebras even though there is no finite witness for their behavioural equivalence. Hence, to obtain the semantics of finite systems, the final coalgebra needs to be replaced by another semantic domain that properly captures finite behaviours.

One prime challenge in systematically developing a theory of finite behaviours is that category theory offers several natural ways of modelling finite objects in categories, e.g. as finitely presentable, finitely generated, or perfectly presentable objects. These concepts are

* This work is supported by Deutsche Forschungsgemeinschaft (DFG), project AD 187/2-1.

equivalent in the category of sets (and give precisely the finite sets) but generally differ, e.g. in categories of algebraic structures. The first categorical treatment of finite coalgebras and their final semantics was given in the work of Adámek, Milius, and Velebil [2] and later elaborated by Milius [16]. These authors modelled finite systems as coalgebras in a locally finitely presentable category carried by a *finitely presentable* object of states and introduced the *rational fixpoint* of a functor. The latter is essentially a "finitely presentable" version of the final coalgebra: it forms the final locally finitely presentable coalgebra, i.e. every coalgebra with a finitely presentable carrier has a unique homomorphism into it, and two states of such a coalgebra are identified by the unique homomorphism if and only if there is a finitely presentable witness for their behavioural equivalence. In the recent work of Milius, Pattinson, and Wißmann [19], a similar and largely parallel theory is developed for coalgebras based on *finitely generated* (instead of finitely presentable) objects of states: the authors introduce the *locally finite fixpoint* of a functor and prove it to be the final locally finitely generated coalgebra. The locally finite fixpoint has some technical advantages over the rational fixpoint, e.g. it is always a subcoalgebra of the final coalgebra, and it captures many instances of finite behaviours such as regular languages, context-free languages, and algebraic trees.

The goal of the present paper is to propose a uniform account that captures both the classical final semantics and its finite versions in the literature as special cases. For this purpose, we introduce the concept of an $(\mathbb{I}, \mathcal{M})$-*accessible* category, a generalisation of accessible and locally finitely presentable categories [6]. The two parameters $\mathbb{I}$ (a class of diagram schemes) and $\mathcal{M}$ (a class of morphisms) give rise to the notion of an $(\mathbb{I}, \mathcal{M})$-*presentable* object in a category that we take as our general definition of "finite object". By taking different choices of $\mathbb{I}$ and $\mathcal{M}$, the $(\mathbb{I}, \mathcal{M})$-presentable objects of a category instantiate e.g. to finitely presentable, finitely generated, perfectly presentable, or arbitrary objects. We then develop the theory of coalgebras based on $(\mathbb{I}, \mathcal{M})$-presentable objects: under suitable conditions on the categories and coalgebraic type functors, we show the existence of a final locally $(\mathbb{I}, \mathcal{M})$-presentable coalgebra, providing a general finite version of final semantics and putting a common roof over the earlier work in [2, 16, 19]. As a new instance of our setting, we investigate coalgebras in algebraic categories carried by finitely generated free algebras of states. Such coalgebras arise naturally as determinisations of coalgebras with side effects, using the generalised powerset construction of Silva, Bonchi, Bonsangue, and Rutten [22].

Besides providing semantics of state-based systems, final coalgebras are also known to allow for an abstract and elegant category-theoretic approach to the semantics of guarded recursive specifications. Generalising classical work of Elgot [11, 12], Nelson [20] and Tiuryn [23] on algebraic properties of infinite trees, Milius [15] observed that the final coalgebra for a functor is also its initial completely iterative algebra. Analogous characterisations are known for the final locally finitely presentable coalgebra [2] and the final locally finitely generated coalgebra [19], where guarded recursive specifications restricted to a finitely presentable (resp. finitely generated) object of variables are considered. In Section 5, we will establish these results uniformly at our level of generality.

2 Preliminaries

We start by reviewing some concepts from category theory that we use in the paper.

2.1 Filtered colimits and locally finitely presentable categories

A category I is *filtered* if I-colimits commute with finite limits in **Set**. Equivalently, the following two properties hold: (i) for any two objects $X, Y \in I$, there exist morphisms $f\colon X \to Z$ and $g\colon Y \to Z$ with a common codomain Z, and (ii) for any two parallel morphisms $f, g\colon X \to Y$ in I, there exists a morphism $h\colon Y \to Z$ with $h \cdot f = h \cdot g$. A *filtered colimit* in a category $\mathcal{A}$ is a colimit over a diagram $D\colon I \to \mathcal{A}$ with I filtered. If all colimit injections (and thus also all connecting maps of the diagram) are monomorphisms, a filtered colimit is called a *directed union*. An object A of $\mathcal{A}$ is called *finitely presentable* if the hom-functor $\mathcal{A}(A, -)\colon \mathcal{A} \to \mathbf{Set}$ preserves filtered colimits, and *finitely generated* if it preserves directed unions. The full subcategories $\mathcal{A}_{fp}$ and $\mathcal{A}_{fg}$ of all finitely presentable (resp. finitely generated) objects of $\mathcal{A}$ are closed under finite colimits. Moreover, $\mathcal{A}_{fg}$ is closed under strong quotients, i.e. quotients carried by strong epimorphisms. The category $\mathcal{A}$ is *locally finitely presentable* if it is cocomplete, $\mathcal{A}_{fp}$ is essentially small (that is, its objects taken up to isomorphism form a set), and every object can be expressed as a filtered colimit of finitely presentable objects. This implies that also every object is a directed union of finitely generated objects. Examples of locally finitely presentable categories include the category of sets, the category of posets, and every variety of (many-sorted) algebras, e.g. groups, rings, vector spaces, or graphs. The finitely generated objects of a variety are the algebras with finitely many generators, and the finitely presentable objects are the algebras presentable with finitely many generators and relations. See [6] for more on locally finitely presentable categories.

2.2 Sifted colimits and algebraic categories

A category I is *sifted* if I-colimits commute with finite products in **Set**. Every filtered category is sifted, as is every category with binary coproducts. A *sifted colimit* in a category $\mathcal{A}$ is a colimit over a diagram $D\colon I \to \mathcal{A}$ with I sifted. An object A of $\mathcal{A}$ is *perfectly presentable* if the hom-functor $\mathcal{A}(A, -)\colon \mathcal{A} \to \mathbf{Set}$ preserves sifted colimits. The category $\mathcal{A}$ is called *algebraic* if it is cocomplete, its full subcategory $\mathcal{A}_{pp}$ of perfectly presentable objects is essentially small, and every object can be expressed as a sifted colimit of perfect presentable objects. Algebraic categories can be characterised as the categories of models of Lawvere theories; in particular, every variety of (many-sorted) algebras is algebraic. The perfectly presentable objects of an algebraic category are exactly the split quotients of finitely generated free algebras. Sifted colimits in varieties are formed on the level of underlying sets. See [7] for more on algebraic categories.

2.3 Factorisation systems

A *factorisation system* in a category $\mathcal{A}$ is a pair $(\mathcal{E}, \mathcal{M})$, where $\mathcal{E}$ and $\mathcal{M}$ are classes of morphims such that (i) both $\mathcal{E}$ and $\mathcal{M}$ are closed under composition and contain all isomorphisms, (ii) every morphism f of $\mathcal{A}$ has a factorisation $f = m \cdot e$ with $e \in \mathcal{E}$ and $m \in \mathcal{M}$, and (iii) the diagonal fill-in property holds: given morphisms e, f, g, m with $e \in \mathcal{E}$, $m \in \mathcal{M}$ and $f \cdot e = m \cdot g$, there exists a unique morphism d with $d \cdot e = g$ and $m \cdot d = f$. We mention the two important properties of factorisation systems:

1. For any two morphisms m and n with $n \cdot m, n \in \mathcal{M}$ one has $m \in \mathcal{M}$.
2. If $(a_i\colon A_i \to A)_{i\in I}$ is a colimit in $\mathcal{A}$ and $A_i \xrightarrow{e_i} \overline{A}_i \xrightarrow{m_i} A$ is the $(\mathcal{E}, \mathcal{M})$-factorisation of a_i, then also $(m_i\colon \overline{A}_i \to A)_{i\in I}$ is a colimit, provided that all morphisms in $\mathcal{E}$ are epic.

Factorisation systems are discussed in detail in [1].

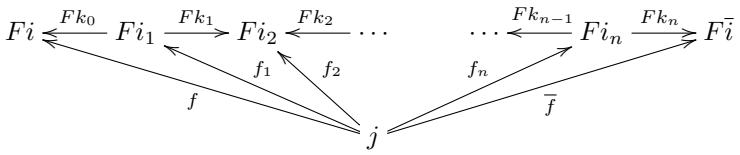
2.4 Coalgebras and algebras

A *coalgebra* for an endofunctor $H\colon \mathcal{A} \to \mathcal{A}$ is a pair (C, γ) of an object $C \in \mathcal{A}$ and a morphism $\gamma\colon C \to HC$. A *homomorphism* between coalgebras (C, γ) and (D, δ) is a morphism $h\colon C \to D$ in $\mathcal{A}$ with $\delta \cdot h = Hh \cdot \gamma$. We denote the category of coalgebras and homomorphisms by $\mathbf{Coalg}\,(H)$. The forgetful functor from $\mathbf{Coalg}\,(H)$ to $\mathcal{A}$ creates colimits, that is, colimits of coalgebras are formed in the underlying category $\mathcal{A}$. If $\mathcal{A}$ has a factorisation system $(\mathcal{E}, \mathcal{M})$ and H preserves $\mathcal{M}$ (i.e. $m \in \mathcal{M}$ implies $Hm \in \mathcal{M}$), then $\mathbf{Coalg}\,(H)$ has the factorisation system of $\mathcal{E}$-carried and $\mathcal{M}$-carried homomorphisms.

Dually, an *algebra* for an endofunctor $H\colon \mathcal{A} \to \mathcal{A}$ is a pair (A, α) of an object $A \in \mathcal{A}$ and a morphism $\gamma\colon HA \to A$. A *homomorphism* between algebras (A, α) and (B, β) is a morphism $h\colon A \to B$ in $\mathcal{A}$ with $h \cdot \alpha = \beta \cdot Hh$.

2.5 Final functors

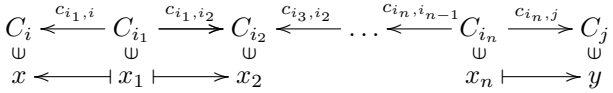
A functor $F\colon I \to J$ is *final* if (i) for every $j \in J$ there exists a morphism $f\colon j \to Fi$ for some $i \in I$, and (ii) any two morphisms $f\colon j \to Fi$ and $\overline{f}\colon j \to F\overline{i}$ with $i, \overline{i} \in I$ are connected by a zig-zag, i.e. there exist morphisms $k_0, \ldots, k_n \in I$ and $f_1, \ldots, f_n \in J$ making the diagram below commutative:



If F is final and $D\colon I \to \mathcal{A}$ and $D'\colon J \to \mathcal{A}$ are two diagrams in a category $\mathcal{A}$ with $D' \cdot F = D$, then D and D' have the same colimits. More precisely, if $(D'_j \xrightarrow{a_j} A)_{j \in J}$ is a colimit cocone over D', then $(D_i = D'_{Fi} \xrightarrow{a_{Fi}} A)_{i \in I}$ is a colimit cocone over D. Conversely, if $(D_i \xrightarrow{a_i} A)_{i \in I}$ is a colimit cocone over D, then $(D'_j \xrightarrow{D' f_j} D'_{Fi_j} = D_{i_j} \xrightarrow{a_{i_j}} A)_{j \in J}$ is a colimit cocone over D', where $f_j\colon j \to Fi_j$ is an arbitrary morphism in J with $i_j \in I$, which exists by the finality of F.

2.6 Colimits in Set

A cocone $(c_i\colon C_i \to C)_{i \in I}$ in **Set** forms a colimit if and only if (i) the maps c_i are jointly surjective (i.e. every element of C lies in the image of some c_i) and (ii) for any two elements $x \in C_i$ and $y \in C_j$ $(i, j \in I)$ with $c_i(x) = c_j(y)$, there exists a zig-zag of connecting morphisms and elements $x_1, \ldots, x_n$ as shown below:



If I is filtered, (ii) can be simplified as follows: for any $x, y \in C_i$ $(i \in I)$ with $c_i(x) = c_i(y)$, there exists a connecting morphism $c_{ij} : C_i \to C_j$ $(j \in I)$ with $c_{ij}(x) = c_{ij}(y)$.

3 $(\mathbb{I}, \mathcal{M})$-accessible categories

As the foundation for our approach to the semantics of finite behaviours we present in this section a mild generalisation of the concept of a locally finitely presentable category (cf. 2.1).

While in the latter the concept of a "finite" object is given by finitely presentable objects, we introduce two parameters that allow for some flexibility in choosing the desired concept of finiteness. Throughout this paper let us fix a category $\mathcal{A}$ with a factorisation system $(\mathcal{E},\mathcal{M})$, with $\mathcal{E}$ a class of epimorphisms, and a collection $\mathbb{I}$ of (not necessarily small) sifted categories.

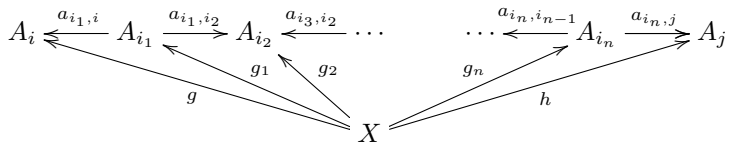
▸ **Definition 3.1.** A category $\mathcal{A}$ is $\mathbb{I}$-*cocomplete* if every $\mathbb{I}$-diagram (i.e. every diagram $I \to \mathcal{A}$ with $I \in \mathbb{I}$) has a colimit. A colimit cocone $(a_i\colon A_i \to A)_{i\in I}$ over an $\mathbb{I}$-diagram is called an $(\mathbb{I},\mathcal{M})$-*colimit* if all injections a_i lie in $\mathcal{M}$. (The latter implies that all connecting morphisms $a_{ij}\colon A_i \to A_j$ lie in $\mathcal{M}$, see 2.3). A functor $F\colon \mathcal{A} \to \mathcal{B}$ *preserves* $(\mathbb{I},\mathcal{M})$-*colimits* if $(Fa_i\colon FA_i \to FA)_{i\in I}$ is a colimit cocone in $\mathcal{B}$ for any $(\mathbb{I},\mathcal{M})$-colimit $(a_i\colon A_i \to A)_{i\in I}$ in $\mathcal{A}$.

▸ **Definition 3.2.** An object X of $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-*presentable* if the hom-functor $\mathcal{A}(X,-)\colon \mathcal{A} \to$ **Set** preserves $(\mathbb{I},\mathcal{M})$-colimits. We denote by $\mathcal{A}_{\mathbb{I},\mathcal{M}}$ the full subcategory of all $(\mathbb{I},\mathcal{M})$-presentable objects.

▸ **Remark 3.3.** Using the characterisation of colimits in **Set**, see 2.6, an object X is $(\mathbb{I},\mathcal{M})$-presentable if and only if for any $(\mathbb{I},\mathcal{M})$-colimit $(a_i\colon A_i \to A)_{i\in I}$ in $\mathcal{A}$ the following holds:

(i) For every morphism $f\colon X \to A$, there exists $i \in I$ and $g\colon X \to A_i$ with $f = a_i \cdot g$.

(ii) For any two morphisms $g\colon X \to A_i$ and $h\colon X \to A_j$ $(i,j \in I)$ with $a_i \cdot g = a_j \cdot h$, there exists a zig-zag of connecting morphisms along with morphisms $g_1,\dots,g_n$ such that the diagram below commutes:



If I is filtered, (ii) simplifies as follows: for any two morphisms $g,h\colon X \to A_i$ $(i \in I)$ with $a_i \cdot g = a_i \cdot h$, there is a connecting morphism a_{ij} $(j \in J)$ with $a_{ij} \cdot g = a_{ij} \cdot h$.

▸ **Definition 3.4.** The category $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-*accessible* if (i) $\mathcal{A}$ is $\mathbb{I}$-cocomplete, and (ii) every object of $\mathcal{A}$ is an $\mathbb{I}$-colimit of $(\mathbb{I},\mathcal{M})$-presentable objects.

▸ **Example 3.5.**

1. Every category $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-accessible w.r.t. the parameters

   $$\mathbb{I} = \text{categories with a terminal object} \quad \text{and} \quad (\mathcal{E},\mathcal{M}) = (\,\text{isomorphisms, all morphisms}\,).$$

   Note that $(\mathbb{I},\mathcal{M})$-colimits are trivial, and thus every object of $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-presentable.
2. Every locally finitely presentable category $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-accessible w.r.t.

   $$\mathbb{I} = \text{small filtered categories} \quad \text{and} \quad (\mathcal{E},\mathcal{M}) = (\,\text{isomorphisms, all morphisms}\,).$$

   Then $(\mathbb{I},\mathcal{M})$-colimits are filtered colimits, and thus $(\mathbb{I},\mathcal{M})$-presentable objects are precisely the finitely presentable ones.
3. Every locally finitely presentable category $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-accessible w.r.t.

   $$\mathbb{I} = \text{small filtered categories} \quad \text{and} \quad (\mathcal{E},\mathcal{M}) = (\,\text{strong epimorphisms, monomorphisms}\,).$$

   Then $(\mathbb{I},\mathcal{M})$-colimits are directed unions, and thus $(\mathbb{I},\mathcal{M})$-presentable objects coincide with the finitely generated ones.

4. Every algebraic category $\mathcal{A}$ is $(\mathbb{I}, \mathcal{M})$-accessible w.r.t.

$$\mathbb{I} = \text{small sifted categories} \quad \text{and} \quad (\mathcal{E}, \mathcal{M}) = (\,\text{isomorphisms, all morphisms}\,).$$

Then $(\mathbb{I}, \mathcal{M})$-colimits are sifted colimits, and thus $(\mathbb{I}, \mathcal{M})$-presentable objects are precisely the perfectly presentable ones.

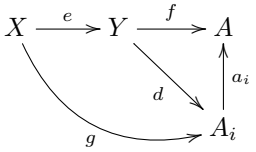
In the following we generalise some important technical properties of locally finitely presentable categories to $(\mathbb{I}, \mathcal{M})$-accessible categories.

▸ **Lemma 3.6.** *$\mathcal{A}_{\mathbb{I},\mathcal{M}}$ is closed under binary coproducts and $\mathcal{E}$-carried quotients.*

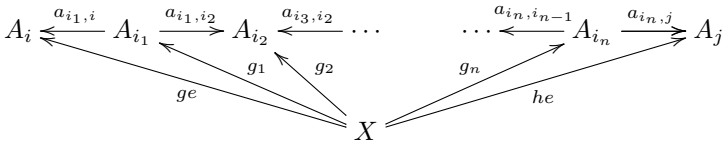
Proof. *Closure under finite coproducts.* Let $X, Y \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$, i.e. the functors $\mathcal{A}(X, -)$ and $\mathcal{A}(Y, -)$ preserve $(\mathbb{I}, \mathcal{M})$-colimits, and suppose that X and Y have a coproduct $X + Y$ in $\mathcal{A}$. Since all categories in $\mathbb{I}$ are sifted by assumption, $\mathbb{I}$-colimits commute with finite products in **Set** (see 2.2). Therefore the functor $\mathcal{A}(X + Y, -) \simeq \mathcal{A}(X, -) \times \mathcal{A}(Y, -)$ preserves $(\mathbb{I}, \mathcal{M})$-colimits, which proves $X + Y \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$.

Closure under $\mathcal{E}$-quotients. Let $e\colon X \to Y$ be a morphism in $\mathcal{E}$ with $X \in \mathcal{A}_{\mathbb{I},M}$, and suppose that $(a_i\colon A_i \to A)_{i\in I}$ is an $(\mathbb{I}, \mathcal{M})$-colimit. To show that $Y \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$, we verify the criterion of Remark 3.3.

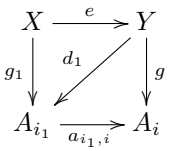
(i) Let $f\colon Y \to A$. Since $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$, the morphism $f \cdot e$ factorises through the cocone, i.e. $f \cdot e = a_i \cdot g$ for some $i \in I$ and $g\colon X \to A_i$. Diagonal fill-in gives a morphism $d\colon Y \to A_i$ with $a_i \cdot d = f$.



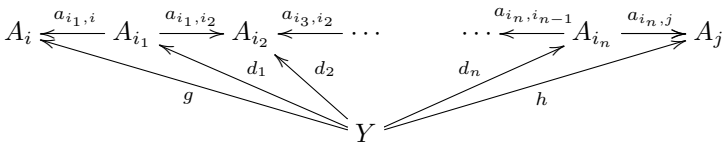
(ii) Let $g\colon Y \to A_i$ and $h\colon Y \to A_j$ be two morphisms with $i, j \in I$ and $a_i \cdot g = a_j \cdot h$. Since $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$, there exists a zig-zag connecting $g \cdot e$ and $h \cdot e$ as in the diagram below:



Since the connecting morphism $a_{i_1,i}$ lies in $\mathcal{M}$, diagonal fill-in yields a morphism $d_1\colon Y \to A_{i_1}$ with $d_1 \cdot e = g_1$ and $a_{i_1,1} \cdot d_1 = g$.



Put $d_2 := a_{i_1,i_2} \cdot d_1$. Again by diagonal fill-in, there exists $d_3\colon X \to A_{i_3}$ with $a_{i_3,i_2} \cdot d_3 = d_2$. Proceeding in this fashion, we obtain morphisms $d_1, \ldots, d_n$ making the following diagram commute:



Thus g and h are connected by a zig-zag, as required. ◂

▸ **Corollary 3.7.** *In an $(\mathbb{I},\mathcal{M})$-accessible category $\mathcal{A}$, every object is an $(\mathbb{I},\mathcal{M})$-colimit of $(\mathbb{I},\mathcal{M})$-presentable objects.*

Proof. Since $\mathcal{A}$ is $(\mathbb{I},\mathcal{M})$-accessible, every object is an $\mathbb{I}$-colimit of $(\mathbb{I},\mathcal{M})$-presentable objects. Therefore the statement follows from 2.3(2) and the closure of $\mathcal{A}_{\mathbb{I},\mathcal{M}}$ under $\mathcal{E}$-quotients. ◂

▸ **Notation 3.8.** For any object $A \in \mathcal{A}$, denote by $\mathcal{A}_{\mathbb{I},\mathcal{M}} \downarrow A$ the comma category whose objects are the morphisms $f\colon X \to A$ with $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$; a morphism from $(f\colon X \to A)$ to $(g\colon Y \to A)$ is a morphism $h\colon X \to Y$ in $\mathcal{A}$ with $f = g \cdot h$. The *canonical diagram* of A is the diagram $\pi_A\colon (\mathcal{A}_{\mathbb{I},\mathcal{M}} \downarrow A) \to \mathcal{A}$ mapping $(f\colon X \to A)$ to X and $h\colon f \to g$ to h.

▸ **Lemma 3.9.** *If $\mathcal{A}$ is an $(\mathbb{I},\mathcal{M})$-accessible category, then every object $A \in \mathcal{A}$ is the colimit of its canonical diagram, with colimit injections $f\colon \pi_A(f) \to A$ $(f \in \mathcal{A}_{\mathbb{I},\mathcal{M}} \downarrow A)$.*

Proof. Express A as an $(\mathbb{I},\mathcal{M})$-colimit $(a_i\colon D_i \to A)_{i\in I}$ of $(\mathbb{I},\mathcal{M})$-presentable objects D_i, see Corollary 3.7. Then the functor $F\colon I \to \mathcal{A}_{\mathbb{I},\mathcal{M}}$ mapping $i \in I$ to the colimit injection $(a_i\colon D_i \to A) \in \mathcal{A}_{\mathbb{I},\mathcal{M}} \downarrow A$ is final; indeed, the two criteria (i) and (ii) in 2.5 state precisely that every object of $\mathcal{A}_{\mathbb{I},\mathcal{M}}$ is $(\mathbb{I},\mathcal{M})$-presentable. Since $D = \pi_A \cdot F$, it follows from 2.5 that the morphisms $f\colon \pi_A(f) \to A$ form a colimit cocone. ◂

4 Locally $(\mathbb{I},\mathcal{M})$-presentable coalgebras

In this section, we consider finite systems modelled as coalgebras based on $(\mathbb{I},\mathcal{M})$-presentable objects and show the existence of a final locally $(\mathbb{I},\mathcal{M})$-presentable coalgebra, which serves as the semantic domain of finite behaviours.

▸ **Assumptions 4.1.** From now on, fix an endofunctor $H\colon \mathcal{A} \to \mathcal{A}$ on an $(\mathbb{I},\mathcal{M})$-accessible category $\mathcal{A}$. Let $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ denote the full subcategory of $\mathbf{Coalg}(H)$ of all coalgebras (A,α) with $A \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$. We assume that (i) $\mathcal{A}$ has binary coproducts, (ii) H preserves $\mathbb{I}$-colimits, (iii) H preserves $\mathcal{M}$ (i.e. $m \in \mathcal{M}$ implies $Hm \in \mathcal{M}$), and (iv) $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H) \in \mathbb{I}$.

▸ **Notation 4.2.** Let $T_H \xrightarrow{\tau} H(T_H)$ be the colimit of the inclusion $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H) \rightarrowtail \mathbf{Coalg}(H)$. (The colimit exists by Assumption 4.1(iv) and because colimits in $\mathbf{Coalg}(H)$ are formed in the underlying category $\mathcal{A}$). We denote the colimit injections by

$$\alpha^{\#}\colon (A,\alpha) \to (T_H,\tau) \qquad ((A,\alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)).$$

▸ **Example 4.3.** The following settings of categories and functors satisfy our assumptions.

1. Let $\mathcal{A}$ be a category with binary coproducts and $H\colon \mathcal{A} \to \mathcal{A}$ a functor with a final coalgebra. Choose $\mathbb{I}$ = categories with a terminal object and $(\mathcal{E},\mathcal{M})$ trivial as in Example 3.5.1. Then the above assumptions (i)-(iv) are clearly satisfied. The coalgebra T_H is the colimit of all H-coalgebras, i.e. the final coalgebra νH of H.
2. Let $\mathcal{A}$ be a locally finitely presentable category and $H\colon \mathcal{A} \to \mathcal{A}$ a functor preserving filtered colimits (a *finitary* functor for short). Choose $\mathbb{I}$ = small filtered categories and $(\mathcal{E},\mathcal{M})$ trivial as in Example 3.5.2. Then (i), (ii) and (iii) are clearly true, and (iv) holds because finitely presentable objects are stable under finite colimits and colimits of H-coalgebras are formed in $\mathcal{A}$. The coalgebra T_H is the colimit of all H-coalgebras with finitely presentable carrier. This coalgebra is the *rational fixpoint* of H introduced in the work of Adámek, Milius, and Velebil [2], and is denoted by ϱH. The term "fixpoint" will be justified in Lemma 4.5 below.

3. Let $\mathcal{A}$ be locally finitely presentable and $H\colon \mathcal{A} \to \mathcal{A}$ a finitary functor preserving monomorphisms. Choose $\mathbb{I}$ = small filtered categories and $(\mathcal{E}, \mathcal{M})$ = (strong epimorphisms, mononomorphisms) as in Example 3.5.3. Then (i), (ii) and (iii) are clear, and (iv) holds because finitely generated objects are stable under finite colimits. The coalgebra T_H is the colimit of all H-coalgebras with finitely generated carrier; this is the *locally finite fixpoint* of H investigated by Milius, Pattinson, and Wißmann [19]. We denote it by ϑH.
4. Let $\mathcal{A}$ be an algebraic category and $H\colon \mathcal{A} \to \mathcal{A}$ a functor preserving sifted colimits. Choose $\mathbb{I}$ = small sifted categories and $(\mathcal{E}, \mathcal{M})$ trivial as in Example 3.5.4. Then (i), (ii) and (iii) are clear, and (iv) holds because perfectly presentable objects are stable under finite coproducts. The coalgebra T_H is formed as the colimit of all coalgebras with perfectly presentable carrier, and is in the following denoted by φH.

▸ **Remark 4.4.** Since H preserves $\mathcal{M}$, the factorisation system of $\mathcal{A}$ lifts to $\mathbf{Coalg}\,(H)$, see 2.4. Consequently, we can express the coalgebra (T_H, τ) as an $(\mathbb{I}, \mathcal{M})$-colimit of coalgebras in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$. Indeed, for each $(A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ factorise

$$\alpha^{\#} \equiv (A, \alpha) \xrightarrow{e_\alpha} (\overline{A}, \overline{\alpha}) \xrightarrow{m_\alpha} (T_H, \tau)$$

in $\mathbf{Coalg}\,(H)$, where $e_\alpha \in \mathcal{E}$ and $m_\alpha \in \mathcal{M}$. Then $m_\alpha = \overline{\alpha}^{\#}$ because $m_\alpha \cdot e_\alpha = \alpha^{\#} = \overline{\alpha}^{\#} \cdot e_\alpha$ (using that $(-)^{\#}$ forms a cocone) and e_α is an epimorphism. Therefore, by 2.3(2), the homomorphisms

$$\overline{\alpha}^{\#}\colon (\overline{A}, \overline{\alpha}) \to (T_H, \tau) \qquad ((A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H))$$

form an $(\mathbb{I}, \mathcal{M})$-colimit cocone in $\mathbf{Coalg}\,(H)$. Given a homomorphism $h\colon (A, \alpha) \to (B, \beta)$ in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$, we denote by $\overline{h}\colon (\overline{A}, \overline{\alpha}) \to (\overline{B}, \overline{\beta})$ the unique homomorphism (obtained via diagonal fill-in) with $\overline{h} \cdot e_\alpha = e_\beta \cdot h$.

▸ **Lemma 4.5** (Lambek Lemma for T_H)**.** *The coalgebra structure $T_H \xrightarrow{\tau} H(T_H)$ is an isomorphism in $\mathcal{A}$.*

Proof sketch. By Remark 4.4 and since colimits in $\mathbf{Coalg}\,(H)$ are formed in $\mathcal{A}$, we know that τ is the unique mediating morphism with $\tau \cdot \overline{\alpha}^{\#} = H\overline{\alpha}^{\#} \cdot \overline{\alpha}$ for all $(A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$. One can show that the morphisms

$$\overline{A} \xrightarrow{\overline{\alpha}} H\overline{A} \xrightarrow{H\overline{\alpha}^{\#}} HT_H \qquad ((A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H))$$

form a colimit cocone over the diagram $D\colon \mathbf{Coalg}_{\mathbb{I},M}(H) \to \mathcal{A}$ mapping (A, α) to $\overline{A}$ and $h\colon (A, \alpha) \to (B, \beta)$ to $\overline{h}$. Then the uniqueness of colimits implies that τ is an isomorphism. The details of the proof are given in the Appendix. ◂

▸ **Example 4.6.**

1. Consider the setting of Example 4.3.1. Then the above lemma is precisely the classical Lambek lemma [14]: the final coalgebra νH is a fixpoint of H.
2. In the setting of Example 4.3.2, the lemma shows that ϱH forms a fixpoint of H. This was shown in [2] with a conceptually different proof method.
3. In the setting of Example 4.3.3, the lemma shows that ϑH is a fixpoint of H. This result is known from [19] where again a different proof method was used.
4. In the setting of Example 4.3.4, we obtain a new fixpoint φH for any sifted colimit preserving endofunctor H on an algebraic category $\mathcal{A}$. The fixpoint φH models the behaviours of *pp-coalgebras*, i.e. coalgebras with perfectly presentable carrier. Given

two pp-coalgebras (A,α) and (B,β), two states $a \in A$ and $b \in B$ are merged by the colimit injections $\alpha^{\#}\colon A \to \varphi H$ and $\beta^{\#}\colon B \to \varphi H$ if and only if there exists a "perfectly presentable reason" for it, in the sense that a and b are connected by a zig-zag of pp-coalgebras as in the diagram below:

$$\begin{array}{ccccccccccc} (A,\alpha) & \xleftarrow{g} & (A_1,\alpha_1) & \xrightarrow{h_1} & (A_2,\alpha_2) & \xleftarrow{h_2} & \dots & \xleftarrow{h_{n-1}} & (A_n,\alpha_n) & \xrightarrow{h} & (B,\beta) \\ \cup & & \cup & & \cup & & & & \cup & & \cup \\ a & \longmapsto & a_1 & \longmapsto & a_2 & & & & a_n & \longmapsto & b \end{array}$$

Indeed, this follows from the fact that the sifted colimit defining φH is formed on the level of **Set**, see 2.2 and 2.6.

One natural occurrence of pp-coalgebras, and in particular coalgebras carried by finitely generated free algebras, arises in the generalised powerset construction of Silva, Bonchi, Bonsangue, and Rutten [22]: given a monad $\mathbf{T} = (T,\eta,\mu)$ on **Set** and an endofunctor $H\colon \mathbf{Set} \to \mathbf{Set}$ that admits a lifting $\overline{H}\colon \mathcal{A}^{\mathbf{T}} \to \mathcal{A}^{\mathbf{T}}$ to the category of $\mathbf{T}$-algebras, a coalgebra $X \to HTX$ for the functor HT can be transformed into a coalgebra $\mathbf{T}X \to \overline{H}\mathbf{T}X$ for the functor $\overline{H}$ whose carrier is the free $\mathbf{T}$-algebra $\mathbf{T}X = (TX,\mu_X)$ on X. For example, the classical powerset construction for nondeterministic automata is an instance of the generalised one by taking $H = \{0,1\} \times \mathsf{Id}^{\Sigma}$ and the finite powerset monad $\mathbf{T} = \mathcal{P}_f$.

▸ **Remark 4.7.** In $\mathcal{A} = \mathbf{Set}$ perfectly presentable, finitely presentable and finitely generated objects coincide with the finite sets, and moreover every finitary set functor H preserves sifted colimits, see [7, Corollary 6.30]. Therefore, for any such H we have $\varphi H = \varrho H = \vartheta H$. For example, if $H = H_\Sigma = \coprod_{\sigma\in\Sigma} \mathsf{Id}^{\mathsf{ar}(\sigma)}$ is the polynomial set functor associated to a finitary signature Σ, then the final coalgebra νH_Σ is carried by the set of finite or infinite Σ-trees, and $\varrho H_\Sigma = \vartheta H_\Sigma = \varphi H_\Sigma$ is carried by the set of *rational trees* [13], i.e. finite or infinite trees that up to isomorphism have only finitely many subtrees. We shall see in Example 4.12 below that in general algebraic categories, the fixpoint φH may differ from ϱH and ϑH.

Our next goal is to characterise the coalgebra T_H by a universal property.

▸ **Definition 4.8.** An H-coalgebra is called *locally* $(\mathbb{I},\mathcal{M})$*-presentable* if it is an $\mathbb{I}$-colimit of coalgebras in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$.

▸ **Remark 4.9.** By factorising as in Remark 4.4, it follows that a locally $(\mathbb{I},\mathcal{M})$-presentable coalgebra can also be expressed as an $(\mathbb{I},\mathcal{M})$-colimit of coalgebras in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$.

▸ **Lemma 4.10.** *If all categories in* $\mathbb{I}$ *are filtered, then every coalgebra in* $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ *is an* $(\mathbb{I},\mathcal{M})$*-presentable object of* $\mathbf{Coalg}(H)$.

For the case where H is a finitary functor on a locally finitely presentable category $\mathcal{A}$ (the setting of Example 4.3.2), this is shown in [8, Lemma III.2]. The following proof is a straightforward generalisation.

Proof. Let $(b_i\colon (B_i,\beta_i) \to (B,\beta))_{i\in I}$ be an $(\mathbb{I},\mathcal{M})$-colimit in $\mathbf{Coalg}(H)$, and suppose that $h\colon (A,\alpha) \to (B,\beta)$ is a coalgebra homomorphism with $(A,\alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$. We need to show that h factorises through the cocone essentially uniquely. Since the colimit is formed in $\mathcal{A}$ and the object A is $(\mathbb{I},\mathcal{M})$-presentable, there exists $i \in I$ and a morphism $g\colon A \to B_i$ in $\mathcal{A}$ with $h = b_i \cdot g$. Moreover, since H preserves $\mathbb{I}$-colimits we have that also $(Hb_i)_{i\in I}$ is a colimit cocone. The two morphisms $\beta_i \cdot g, Hg \cdot \alpha\colon A \to HB_i$ are merged by Hb_i because

$$Hb_i \cdot \beta_i \cdot g = \beta \cdot b_i \cdot g = \beta \cdot h = Hh \cdot \alpha = Hb_i \cdot Hg \cdot \alpha,$$

using that h and b_i are coalgebra homomorphisms. Therefore, since I is filtered, there exists a connecting morphism $b_{ij}\colon (B_i,\beta_i) \to (B_j,\beta_j)$ with $Hb_{ij}\cdot\beta_i\cdot g = Hb_{ij}\cdot Hg\cdot\alpha$. An easy computation now shows that the morphism $f := b_{ij}\cdot g$ is a coalgebra homomorphism from (A,α) to (B_j,β_j) with $h = b_j\cdot f$. Thus h factorises through b_j in $\mathbf{Coalg}\,(H)$.

The uniqueness of factorisations is clear because this holds in the underlying category. ◂

▸ **Theorem 4.11.** *If all categories in $\mathbb{I}$ are filtered, then (T_H,τ) is the final locally $(\mathbb{I},\mathcal{M})$-presentable H-coalgebra.*

Proof. By definition, the coalgebra (T_H,τ) is locally $(\mathbb{I},\mathcal{M})$-presentable. To show the finality, it suffices to prove that every coalgebra (A,α) in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ has a unique homomorphism into (T_H,τ). Clearly the colimit injection $\alpha^\#\colon (A,\alpha)\to(T_H,\tau)$ is a homomorphism. For the uniqueness, suppose that $h\colon (A,\alpha)\to(T_H,\tau)$ is any homomorphism. Since (A,α) is $(\mathbb{I},\mathcal{M})$-presentable in $\mathbf{Coalg}\,(H)$ by Lemma 4.10 and the homomorphisms $\overline{\beta}^\#\colon(\overline{B},\overline{\beta})\to(T_H,\tau)$ $((B,\beta)\in\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H))$ form an $(\mathbb{I},\mathcal{M})$-colimit cocone by Remark 4.4, there exists a coalgebra (B,β) in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ and a homomorphism $g\colon(A,\alpha)\to(\overline{B},\overline{\beta})$ with $\overline{\beta}^\#\cdot g = h$. Since the morphisms $(-)^\#$ form a cocone, this implies $h = \overline{\beta}^\#\cdot g = \alpha^\#$. ◂

▸ **Example 4.12.** In the setting of Example 4.3.4 where the categories in $\mathbb{I}$ are not filtered, the universal property in the above theorem generally fails, that is, a pp-coalgebra can admit more than one homomorphism into φH. To see this, consider the category $\mathcal{A}$ of algebras with a single unary operation u and the identity functor $H = \mathsf{Id}$ on $\mathcal{A}$. Thus H-coalgebras are input-free deterministic transition systems endowed with an additional unary operation that commutes with the transitions. Let FX denote the free algebra of $\mathcal{A}$ over the set X, carried by the set of all terms $u^n(x)$ with $n\geqslant 0$ and variables $x\in X$. Note that split quotients of a term algebra are again term algebras (arising by identifiying variables). Therefore the perfectly presentable objects of $\mathcal{A}$ are exactly the finitely generated free algebras, i.e. the algebras FX with X finite (cf. 2.2). We write $\mathbf{Coalg}_{\text{free}}(H)$ for the category of all H-coalgebras with finitely generated free carrier.

Consider the two H-coalgebras

$$F\{x\}\xrightarrow{\alpha}F\{x\}\quad\text{with}\quad\alpha(x)=x$$

and

$$F\{y\}\xrightarrow{\beta}F\{y\}\quad\text{with}\quad\beta(y)=u(y),$$

and let $g\colon F\{x\}\to\varphi H$ be the unique morphism in $\mathcal{A}$ with $g(x)=\beta^\#(y)$. We will prove that (i) $g\colon(F\{x\},\alpha)\to(\varphi H,\tau)$ is a coalgebra homomorphism and (ii) $g\neq\alpha^\#$, which shows that there are two distinct homomorphisms from $(F\{x\},\alpha)$ into $(\varphi H,\tau)$.

To prove (i), observe first that clearly $\beta\colon(F\{y\},\beta)\to(F\{y\},\beta)$ is a coalgebra homomorphism, and thus $\beta^\#=\beta^\#\cdot\beta$ because $(-)^\#$ forms a cocone. This implies

$$\begin{aligned}
g(\alpha(x)) &= g(x) && \text{(def. } \alpha)\\
&= \beta^\#(y) && \text{(def. } g)\\
&= \beta^\#(\beta(y)) && \text{(see above)}\\
&= \tau(\beta^\#(y)) && (\beta^\#\text{ coalg. hom.})\\
&= \tau(g(x)) && \text{(def. } g)
\end{aligned}$$

and thus $g\cdot\alpha=\tau\cdot g$ because x generates $F\{x\}$. Hence g is a coalgebra homomorphism.

To prove (ii), it suffices to show that $\alpha^{\#}(x) \neq \beta^{\#}(y)$. Since the sifted colimit defining φH is formed in **Set** (see 2.2), this requires us to show that there is no zig-zag of coalgebra homomorphisms in $\mathbf{Coalg}_{\text{free}}(H)$ connecting x and y. In the following, let us call an element a of an H-coalgebra (A, γ) *finite* if the set $\{\gamma^n(a) : n \geqslant 0\}$ of all states that are reachable from a by transitions is finite.

$(*)$ *Claim.* Let X and Y be finite sets and let $h\colon (FX, \gamma) \to (FY, \delta)$ be a coalgebra homomorphism. Then a state $t \in FX$ is finite if and only if the state $h(t) \in FY$ is finite.

Proof. Since h is a coalgebra homomorphism we have $h(\gamma^n(t)) = \delta^n(h(t))$ for every $n \geqslant 0$. This immediately implies that $h(t)$ is finite whenever t is finite. Conversely, suppose that t is not finite. Since the set X of variables is finite, there are only finitely many terms of any given height in FX. Thus, for every $k \geqslant 0$, there exists a term $u^n(x) \in TX$ with $x \in X$ and $n \geqslant k$ that is reachable from t. Then $h(u^n(x)) = u^n(h(x))$ is a term of height at least $n \geqslant k$ in FY, and this term is reachable from $h(t)$ because h is a coalgebra homomorphism. Thus the state $h(t) \in FY$ is not finite. ◂

Since the state x of the coalgebra $(F\{x\}, \alpha)$ is finite and the state y of $(F\{y\}, \beta)$ is infinite, $(*)$ shows that no zig-zag in $\mathbf{Coalg}_{\text{free}}(H)$ connecting x and y exists.

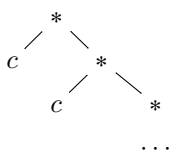
▸ **Remark 4.13.** In the above example the fixpoints ϱH, ϑH and νH are carried by the terminal object 1, while φH is nontrivial. In general, all four fixpoints may be pairwise distinct. This holds, e.g., for the endofunctor $H = \mathbb{N} \times \mathsf{Id}$ on the category $\mathcal{A}$ of sets with two unary operations, with both operations on $\mathbb{N}$ given by the successor map; see Milius [17]. In addition, in *loc. cit.* the author discusses sufficient conditions on the functor H ensuring that the three fixpoints φH, ϱH and ϑH (each of which represents different flavours of finite behaviours) coincide.

5 $(\mathbb{I}, \mathcal{M})$-iterative algebras

In this section, we establish another universal property of T_H: it is the initial $(\mathbb{I}, \mathcal{M})$-iterative algebra and thus forms the universal domain of solutions for guarded recursive specifications. The results of this section put a common roof over results from [15, 2, 19]. Since the proofs are essentially identical to the ones of [2], we confine ourselves to describing the constructions involved.

For motivation, recall that for an algebra A over a finitary signature Σ, a *flat system of equations* is a finite system of recursive equations of the form $x_1 = t_1, \dots, x_n = t_n$ where $x_1, \dots x_n$ are the variables and each t_i is either an element of A or a Σ-term of height 1 in the variables $x_1, \dots, x_n$. Thus, a flat system corresponds to a function $e\colon X \to H_\Sigma X + A$ with $X = \{x_1, \dots x_n\}$. The algebra A is called *iterative* if every flat system of equations has a unique solution in A.

▸ **Example 5.1.** The Σ-algebra of finite or infinite trees is iterative, as is the Σ-algebra of rational trees (see Remark 4.7). Given the signature Σ of a binary operation symbol $*$ and a constant symbol c, the flat system $x_1 = x_2 * x_1$, $x_2 = c$, has the following unique solution in the algebra of rational trees:

$x_1 =$ 

$x_2 = c.$

Replacing H_Σ by a general endofunctor $H\colon \mathcal{A} \to \mathcal{A}$ and the finite set X of variables an arbitrary $(\mathbb{I}, \mathcal{M})$-presentable object, the concept of an iterative Σ-algebra generalises has the following categorical generalisation:

▸ **Definition 5.2.** Let $HA \xrightarrow{\alpha} A$ be an H-algebra. By a *flat equation morphism* is meant a morphism $e\colon X \to HX + A$ with $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$. A *solution* of e in (A, α) is a morphism $e^\dagger\colon X \to A$ making the following square commute:

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array}$$

The algebra (A, α) is called $(\mathbb{I}, \mathcal{M})$-*iterative* if every flat equation morphism admits a unique solution.

▸ **Example 5.3.** In the settings of Example 4.3.1-3, $(\mathbb{I}, \mathcal{M})$-iterative algebras are called *completely iterative algebras* [15], *iterative algebras* [2] and *fg-iterative algebras* [19], respectively.

Since the coalgebra structure τ of T_H is an isomorphism by Lemma 4.5, we can view T_H as an H-algebra (T_H, τ^{-1}). We aim to show that this algebra is $(\mathbb{I}, \mathcal{M})$-iterative and, in fact, the initial $(\mathbb{I}, \mathcal{M})$-iterative algebra. This requires further assumptions on our setting:

▸ **Assumptions 5.4.** In addition to the Assumptions 4.1 given in the previous section, we assume that (i) all categories in $\mathbb{I}$ are filtered, (ii) $\mathcal{M}$ is closed under coproducts, i.e. $m, m' \in \mathcal{M}$ implies $m + m' \in \mathcal{M}$, and (iii) $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X) \in \mathbb{I}$ for every $X \in \mathcal{A}$.

Here $H + X\colon \mathcal{A} \to \mathcal{A}$ is the endofunctor given by $Y \mapsto HY + X$.

▸ **Example 5.5.** In the setting of Example 4.3.1, the above assumption (iii) states precisely that H is an *iteratable endofunctor*, i.e. the functor $H + X$ admits a final coalgebra for every $X \in \mathcal{A}$. In Example 4.3.2/3, (iii) is trivially satisfied.

▸ **Lemma 5.6.** *(T_H, τ^{-1}) is an $(\mathbb{I}, \mathcal{M})$-iterative algebra.*

Proof sketch. Let $e\colon X \to HX + T_H$ be a flat equation morphism with $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$. Express the coalgebra (T_H, τ) as an $(\mathbb{I}, \mathcal{M})$-colimit

$$\overline{\alpha}^\#\colon (\overline{A}, \overline{\alpha}) \to (T_H, \tau) \qquad ((A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H),$$

see Remark 4.4. Since colimits commute with coproducts and $\mathcal{M}$ is closed under coproducts, we have the $(\mathbb{I}, \mathcal{M})$-colimit

$$HX + \overline{\alpha}^\#\colon HX + \overline{A} \to HX + T_H \qquad ((A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H))$$

in $\mathcal{A}$. Therefore, since X is $(\mathbb{I}, \mathcal{M})$-presentable, there exists a coalgebra $(A, \alpha) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ and a morphism e_0 making the triangle below commute:

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + T_H \\ & {\scriptstyle e_0}\searrow & \uparrow{\scriptstyle HX + \overline{\alpha}^\#} \\ & & HX + \overline{A} \end{array}$$

Form the coalgebra

$$s \equiv X + \overline{A} \xrightarrow{[e_0,\mathsf{inr}]} HX + \overline{A} \xrightarrow{HX+\overline{\alpha}} HX + H\overline{A} \xrightarrow{\mathsf{can}} H(X + \overline{A})$$

where inl and inr denote the left and right coproduct injections, and can is the canonical morphism determined by $\mathsf{can} \cdot \mathsf{inl} = H\mathsf{inl}$ and $\mathsf{can} \cdot \mathsf{inr} = H\mathsf{inr}$. Letting $s^\#$ be the unique homomorphism into (T_H, τ), the morphism

$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + \overline{A} \xrightarrow{s^\#} T_H$$

can be shown to be the unique solution of e. The argument is identical to the proof of [2, Lemma 3.5]. ◂

▸ **Theorem 5.7.** *(T_H, τ^{-1}) is the initial $(\mathbb{I}, \mathcal{M})$-iterative algebra.*

Proof sketch. Let $HA \xrightarrow{\alpha} A$ be an $(\mathbb{I}, \mathcal{M})$-iterative algebra. Any coalgebra $(X, \xi) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$ induces a flat equation morphism

$$e_\xi \equiv X \xrightarrow{\xi} HX \xrightarrow{\mathsf{inl}} HX + A$$

with the unique solution $e_\xi^\dagger \colon X \to A$. The morphisms $e_\xi^\dagger$ form a cocone in $\mathcal{A}$ over the diagram defining T_H. Therefore there exists a unique $h \colon T_H \to A$ in $\mathcal{A}$ with $h \cdot \xi^\# = e_\xi^\dagger$ for all $(X, \xi) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H)$, which can be shown to be the unique H-algebra homomorphism from (T_H, τ^{-1}) to (A, α). The proof is analogous to [2, Theorem 3.3] ◂

▸ **Example 5.8.** By specialising the above theorem to the settings of Example 4.3.1-3, we recover the following three results from the literature [15, 2, 19]:

1. If $\mathcal{A}$ is a category with binary coproducts and H is a functor with a final coalgebra νH, then νH is the initial completely iterative algebra for H.
2. If $\mathcal{A}$ is locally finitely presentable and H is a finitary functor, then ϱH is the initial iterative algebra for H.
3. If $\mathcal{A}$ is locally finitely presentable with coproducts stable under monomorphisms, and H is a finitary functor preserving monomorphisms, then ϑH is the initial fg-iterative algebra for H.

▸ **Remark 5.9** (Free $(\mathbb{I}, \mathcal{M})$-iterative algebras)**.** The forgetful functor from the category of all $(\mathbb{I}, \mathcal{M})$-iterative algebras and homomorphisms into $\mathcal{A}$ has a left adjoint. The free $(\mathbb{I}, \mathcal{M})$-iterative algebra over an object $X \in \mathcal{A}$ is constructed as follows.

Observe first that the functor $H + X$ satisfies the Assumptions 4.1: it preserves $\mathbb{I}$-colimits because H does and colimits commute with coproducts; it preserves $\mathcal{M}$ because H does and $\mathcal{M}$ is stable under coproducts by Assumption 5.4(ii); and one has $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X) \in \mathbb{I}$ by Assumption 5.4(iii). Therefore Theorem 4.11 (applied to the functor $H + X$ in lieu of H) shows that there exists a final locally $(\mathbb{I}, \mathcal{M})$-presentable coalgebra

$$T_H X := T_{H+X}$$

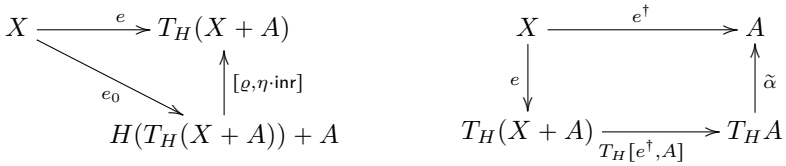
for $H + X$, constructed as the colimit of all coalgebras in $\mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X)$. We denote the coalgebra structure of $T_H X$ and its inverse (see Lemma 4.5) by

$$T_H X \xrightarrow{\tau_X} H(T_H X) + X \quad \text{and} \quad H(T_H X) + X \xrightarrow{[\varrho_X, \eta_X]} T_H X,$$

respectively. The latter is the initial $(\mathbb{I}, \mathcal{M})$-iterative algebra for $H + X$ by Theorem 5.7. Then a standard argument identical to [15, Theorem 2.10] shows that $H(T_H X) \xrightarrow{\varrho_X} T_H X$ is the free $(\mathbb{I}, \mathcal{M})$-iterative H-algebra over X, with unit $\eta_X \colon X \to T_H X$.

By definition, $(\mathbb{I}, \mathcal{M})$-iterative algebras have unique solutions for every flat equation morphism. This property implies a much stronger one: every *guarded* equation morphism has a unique solution. Recall that for a Σ-algebra A, a *guarded system of equations* consists of equations $x_1 = t_1, \ldots, x_n = t_n$ where each t_i is either an element of A or a rational Σ-tree over $X + A$ of height at least 1. This concept can be generalised to our present setting as follows:

▸ **Definition 5.10.** Let (A, α) be an $(\mathbb{I}, \mathcal{M})$-iterative algebra. By a *guarded equation morphism* is meant a morphism $e\colon X \to T_H(X + A)$ with $X \in \mathcal{A}_{\mathbb{I},\mathcal{M}}$ for which there exists a morphism e_0 making the left-hand triangle below commute. A *solution* of e is a morphism $e^\dagger\colon X \to A$ making the right-hand diagram commute. Here $\widetilde{\alpha}$ is the unique homomorphism with $\widetilde{\alpha} \cdot \eta_A = \mathsf{id}_A$, using the freeness of $T_H A$.



▸ **Theorem 5.11.** *Every $(\mathbb{I}, \mathcal{M})$-iterative algebra admits unique solutions of guarded equation morphisms.*

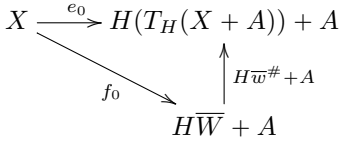
Proof sketch. Let (A, α) be an $(\mathbb{I}, \mathcal{M})$-iterative algebra, and suppose that $e\colon X \to T_H(X+A)$ is a guarded equation morphism with associated $e_0\colon X \to HT_H(X + A) + A$. Express the coalgebra $T_H(X + A)$ as an $(\mathbb{I}, \mathcal{M})$-colimit

$$\overline{w}^{\#}\colon W \to T_H(X + A) \qquad ((W, w) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X + A)),$$

see Remark 4.4. Since H preserves $(\mathbb{I}, \mathcal{M})$-colimits, $\mathcal{M}$ is stable under coproducts and colimits commute with coproducts, it follows that

$$H\overline{w}^{\#} + A\colon H\overline{W} + A \to H(T_H(X + A)) + A \qquad ((W, w) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X + A))$$

forms an $(\mathbb{I}, \mathcal{M})$-colimit cocone in $\mathcal{A}$. Therefore, since X is $(\mathbb{I}, \mathcal{M})$-presentable, the morphism e_0 factorises through $H\overline{w}^{\#} + A$ for some $(W, w) \in \mathbf{Coalg}_{\mathbb{I},\mathcal{M}}(H + X + A)$:



Form the following flat equation morphism, where inm is the middle coproduct injection:

$$s \equiv \overline{W} + X \xrightarrow{[\overline{w},\mathsf{inm}]} H\overline{W} + X + A \xrightarrow{[\mathsf{inl},f_0,\mathsf{inr}]} H\overline{W} + A \xrightarrow{H\mathsf{inl}+A} H(\overline{W} + X) + A$$

Since the algebra (A, α) is $(\mathbb{I}, \mathcal{M})$-iterative, s has the unique solution $s^\dagger\colon X \to A$, and one can verify that

$$e^\dagger \equiv X \xrightarrow{\mathsf{inr}} \overline{W} + \overline{X} \xrightarrow{s^\dagger} A$$

is the unique solution of e. The argument is identical to the proof of [2, Theorem 4.6] ◂

6 Conclusion and Future Work

Our paper has provided the first steps towards a uniform categorical treatment of finite systems and finite recursive specifications, where the meaning of "finite" becomes a parameter that can be chosen according to the applications in mind. As our main technical result we showed that, under suitable assumptions on the categories and functors, there exists a final locally $(\mathbb{I}, \mathcal{M})$-presentable coalgebra that forms a fixpoint of the type functor and captures precisely the behaviours of finite systems. The uniformity of our setting does away with the previous need of developing coalgebraic semantics for each of the competing notions of finiteness independently, often with structurally very similar results and proofs.

In the case of finitary endofunctors on locally finitely presentable categories, the rational fixpoint and its characterisation as the initial iterative algebra formed the starting point for extensive research on iterative monads [5, 4], iteration theories [3], recursive program schemes [18], and proof systems for language equivalence [16, 10] from a (co-)algebraic perspective. We expect that many of the results in *loc. cit.* generalise to our present setting, and thus could be extended to finiteness conditions that so far have not been investigated, e.g. to finitely generated objects of variables.

Acknowledgements. I am grateful to Jiří Adámek and Stefan Milius for many helpful discussions.

References

1 Jiří Adámek, Horst Herrlich, and George E. Strecker. *Abstract and Concrete Categories: The Joy of Cats*. Dover Publications, 2nd edition, 2009.

2 Jiří Adámek, Stefan Milius, and Jiří Velebil. Iterative algebras at work. *Math. Structures Comput. Sci.*, 16(6):1085–1131, 2006.

3 Jiří Adámek, Stefan Milius, and Jiří Velebil. Elgot theories: a new perspective on iteration theories. In *Proc. Mathematical Foundations of Programming Science (MFPS XXV)*, volume 249 of *Electron. Notes Theor. Comp. Sci.*, pages 407–427. Elsevier, 2009.

4 Jiří Adámek, Stefan Milius, and Jiří Velebil. Equational properties of iterative monads. *Inform. and Comput.*, 208:1306–1348, 2010. doi:10.1016/j.ic.2009.10.006.

5 Jiří Adámek, Stefan Milius, and Jiří Velebil. On second-order iterative monads. *Theoret. Comput. Sci.*, 412:4969–4988, 2011. doi:10.1016/j.tcs.2011.04.027.

6 Jiří Adámek and Jiří Rosický. *Locally presentable and accessible categories*. Cambridge University Press, 1994.

7 Jiří Adámek, Jiří Rosický, and Enrico Vitale. *Algebraic Theories*. Cambridge University Press, 2011.

8 Jiří Adámek and Hans-E. Porst. On tree coalgebras and coalgebra presentations. *Theor. Comput. Sci.*, 311(1-3):257–283, 2004.

9 Mikołaj Bojańczyk, Bartek Klin, and Sławomir Lasota. Automata theory in nominal sets. *Logical Methods in Computer Science*, Volume 10, Issue 3, 2014.

10 Marcello M. Bonsangue, Stefan Milius, and Alexandra Silva. Sound and complete axiomatizations of coalgebraic language equivalence. *ACM Trans. Comput. Log.*, 14(1:7), 2013.

11 Calvin C. Elgot. Monadic computation and iterative algebraic theories. In H. E. Rose and J. C. Sheperdson, editors, *Logic Colloquium '73*, volume 80, pages 175–230, Amsterdam, 1975. North-Holland Publishers.

12 Calvin C. Elgot, Stephen L. Bloom, and Ralph Tindell. On the algebraic structure of rooted trees. *J. Comput. System Sci.*, 16:362–399, 1978.

13 Susanna Ginali. Regular trees and the free iterative theory. *J. Comput. System Sci.*, 18:228–242, 1979.

14 Joachim Lambek. A fixpoint theorem for complete categories. *Math. Z.*, 103:151–161, 1968.

15 Stefan Milius. Completely iterative algebras and completely iterative monads. *Inform. and Comput.*, 196:1–41, 2005.

16 Stefan Milius. A sound and complete calculus for finite stream circuits. In *Proc. 25th Annual Symposium on Logic in Computer Science (LICS'10)*, pages 449–458. IEEE Computer Society, 2010.

17 Stefan Milius. Proper functors and their rational fixed point. In *Proc. 7th Conference on Algebra and Coalgebra in Computer Science (CALCO'17)*, Leibniz International Proceedings in Informatics (LIPIcs), 2017.

18 Stefan Milius and Lawrence S. Moss. The category theoretic solution of recursive program schemes. *Theoret. Comput. Sci.*, 366:3–59, 2006. Fundamental study.

19 Stefan Milius, Dirk Pattinson, and Thorsten Wißmann. A new foundation for finitary corecursion: The locally finite fixpoint and its properties. In *Proc. 19th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'16)*, volume 9634 of *Lecture Notes Comput. Sci. (ARCoSS)*, pages 107–125, 2016.

20 Evelyn Nelson. Iterative algebras. *Theoret. Comput. Sci.*, 25:67–94, 1983.

21 Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249(1):3–80, 2000.

22 Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Log. Methods Comput. Sci*, 9(1:9), 2013.

23 Jerzy Tiuryn. Unique fixed points vs. least fixed points. *Theoret. Comput. Sci.*, 12:229–254, 1980.

The EfProb Library for Probabilistic Calculations*

Kenta Cho[1] and Bart Jacobs[2]

1 **Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands**
K.Cho@cs.ru.nl

2 **Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands**
bart@cs.ru.nl

Abstract

EfProb is an abbreviation of Effectus Probability. It is the name of a library for probability calculations in Python. EfProb offers a uniform language for discrete, continuous and quantum probability. For each of these three cases, the basic ingredients of the language are states, predicates, and channels. Probabilities are typically calculated as validities of predicates in states. States can be updated (conditioned) with predicates. Channels can be used for state transformation and for predicate transformation. This short paper gives an overview of the use of EfProb.

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases probability, embedded language, effectus theory

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.25

Category Tool Paper

1 Introduction

The EfProb library provides an embbeded language in Python, for probability. The 'Ef' in EfProb stands for 'Effectus', and 'Prob' stand for 'Probability'. An effectus is an abstract (categorical) model that captures the essentials of discrete, continuous and quantum probability and logic [5, 1]. The EfProb library is based on this categorical model, providing a uniform approach to discrete, continuous, and quantum probability. However, in order to be able to use and understand the basic of the EfProb library it is not required to understand the underlying categorical semantics. The EfProb library makes it easy to model various problems in probability theory and to calculate and plot probability mass/density functions.

The aim of this short paper is to give a brief overview of EfProb, mainly by providing examples. The Python files that define EfProb are available online[1] together with an extensive manual that provides much more information. The core of EfProb has reached a reasonable level of stability, but development is still going on, driven both by practical and theoretical considerations.

We envision that the EfProb library could be useful in teaching the basics of probability theory from a unified perspective. At the same time the library could be useful in scientific research as well, in order to quickly model new examples and compute outcomes.

* The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement nr. 320571.

[1] At efprob.cs.ru.nl

The library is not meant for large scale computation, for instance like those in data analytics. In our development of the library we prefer semantical clarity to execution speed. Indeed, the underlying channel-based semantics of EfProb is mathematically clear, in the sense that the library computes probabilities using the standard mathematical formulas, including, for instance, integration. This 'exact' computation is in contrast to approximate approaches via Monte Carlo sampling, which is used by languages such as BLOG [8], Church [3], and Anglican [11].

In practice, however, outcomes of EfProb are approximations too, since the library is based on numerical computation with floating-point numbers. Scaling is an issue in EfProb, exponential in the number of dimensions. In the continuous case, dimensions greater than 3 are already too slow to handle. It is important to keep in mind that the main functionality of the library is computing numerical outcomes, and not, for instance, logical reasoning.

The EfProb library is split into two parts, one for classical and one for quantum probability. The classical part integrates both discrete and continuous probability.

2 A uniform language

This section briefly describes the common language and notation that is used in EfProb for states, predicates, channels, *etc.* The description will be high-level, abstracting away from the different implementations for discrete, continuous and quantum probability.

States

A state captures probabilities for elements of a certain domain. Classically, they are often called (probability) distributions. States in EfProb are closed under products, marginalisation, and convex sum. The product of two states `s`, `t` is written as `s @ t`. It involves the cartesian product of the underlying domains. Given a 'joint' state on a product domain, one can marginalise the state, for which a post-script notation is used. For instance, if `s` is a state on the product of three domains `D1`, `D2`, `D2`, then one can marginalise `s` in several ways:

`s % [1,0,0]`	is the first marginal (projection)
`s % [0,1,0]`	is the second marginal
`s % [0,1,1]`	is the second and third marginal, *etc.*

In general, for a state on the product of n domains, a 'mask' of length n is used with 0's and 1's, where a 1 at position i tells that the i-th domain should be kept in the marginal.

Convex sums of states exist, like in `r1 * s1 + r2 * s2 + r3 * s3` for states `si` and numbers `ri` from $[0, 1]$ that add up to 1. Such convex sums exist in n-ary form.

Specific states are predefined, specifically for discrete/continuous/quantum probability, like uniform states, point states, Poisson, binomial, Gaussian, *etc.*

Predicates

Predicates in probabilistic logic have the structure of an effect module [5]. On a fixed domain `D`, there are truth and falsity predicates `truth(D)` and `falsity(D)`, which are least and greatest element. Also, for each predicate `p` there is an orthosupplement (negation) `~p` and a rescaling `r*p` for a number `r` in the unit interval. There is a partially defined sum `p+q`, which is defined (as predicate) whenever this sum is below truth. Finally there is a sequential conjunction `p & q`. It is commutative in classical probability, but not in quantum probability.

For two predicates `p` on domain `D` and `q` on `E` there is also the parallel conjunction `p @ q` on the product of the underlying domains. This construction is often used for weakening, when either `p` or `q` is truth, in order to move a predicate to a bigger context.

States and predicates

Given a state `s` and a predicate `p` on the same domain, one writes `s >= p` in EfProb for the validity of `p` in `s`. This yields a number in $[0, 1]$. Also one can write `s/p` for the result of conditioning `s` with `p`. The following version of Bayes' rule holds in general: `s/p >= q` equals `(s >= p & q) / (s >= p)`. In classical probability the order of conditioning is irrelevant — that is, `s/p/q` is the same as `s/q/p` — but in quantum probability the order does matter.

Channels

A channel `c` is a 'probabilistic map' from a domain to a codomain, say from `D` to `E`. For a state `s` on `D`, one writes `c >> s` for the state on `E` obtained by state transformation. In the other direction, for a predicate `q` on `E` one obtains predicate `c << q` on `D` by predicate transformation. There is a basic law that tells that the probabilities `(c >> s) >= q` and `s >= (c << q)` are equal. There are operations $*$ and `@` for sequential and parallel composition of channels. They can be used as a basis for an elementary language for writing programs/protocols.

Channels are often used in Bayesian (backwards) learning (see [7]), in the form of an updated (revised) posterior state `s / (c << q)` of a prior state `s` in the light of evidence `q`.

The EfProb library supports more structure and more functions, for instance for random variables and Bayesian networks, expected values and (co)variance, and for disintegration. In the remainder of this note we sketch some examples, in order to give an impression of the possibilities. For more information we refer to the EfProb Manual [6].

3 Discrete probability

A discrete state/probability distribution ω on a finite domain D is a probability mass function $\omega\colon D \to [0, 1]$ that satisfies $\sum_{x \in D} \omega(x) = 1$. EfProb prints such distributions using 'ket' notation, as $\frac{1}{2}|a\rangle + \frac{1}{6}|b\rangle + \frac{1}{3}|c\rangle$, for $\{a, b, c\} \subseteq D$. A predicate on D is a function $p\colon D \to [0, 1]$. Validity $\omega \models p$ is defined as $\sum_{x \in D} \omega(x) \cdot p(x)$. If this number is non-zero the conditioned state $\omega/p\colon D \to [0, 1]$ is given by $(\omega/p)(x) = \frac{\omega(x) \cdot p(x)}{\omega \models p}$. A channel $D \to E$ is a function from D to distributions on E. It is a Kleisli map for the distribution monad that can be understood as a D-indexed collection of states on E. It is represented in EfProb/Python as a stochastic matrix. Then: $(c \gg \omega)(y) = \sum_x \omega(x) \cdot c(x)(y)$ and $(c \ll q)(x) = \sum_y c(x)(y) \cdot q(y)$.

We consider a beginner's example in Bayesian reasoning. The setting is given by some disease with a priori probability of 1%. There is a test for the disease with the following 'sensitivity'. If someone has the disease, then the test is 90% positive; but if someone does not have the disease, there is still a 5% chance that the test is positive. We formalise this situation in EfProb as a prior state and a channel:

```
>>> disease_dom = ['D', '~D']
>>> prior = flip(1/100, disease_dom)
>>> prior
0.01|D> + 0.99|~D>
>>> sensitivity = chan_from_states([flip(9/10), flip(1/20)], disease_dom)
```

We first use the channel to compute the probability that a test for an arbitrary person is positive. This is done via state transformation:

```
>>> sensitivity >> prior
0.0585|True> + 0.942|False>
```

Next we would like to learn the probability of having the disease after a positive test. This 'positive test' predicate is written as `yes_pred`. It is transformed into a predicate on the disease domain via the sensitivity channel, and then used to update the prior state below. We see that after a positive test the disease probability changes from 1% to 15%.

```
>>> posterior = prior / (sensitivity << yes_pred)
>>> posterior
0.154|D> + 0.846|~D>
```

4 Continuous probability

In general, a continuous state/probability distribution consists of a measurable space (X, Σ) with a probability measure $\omega\colon \Sigma \to [0,1]$. In practice such measures are often given by a probability density function (pdf). This approach is followed in EfProb. Thus, a continuous state is given by a domain consisting of an n-dimensional cube D of real numbers, possibly infinite, with a pdf $\omega\colon D \to \mathbb{R}_{\geq 0}$ satisfying $\int_D \omega(x)\,\mathrm{d}x = 1$. A predicate on D is a (measurable) function $p\colon D \to [0,1]$. Validity $\omega \mathrel{>\!\!=} p$ is given by $\int_D \omega(x) \cdot p(x)\,\mathrm{d}x$. The updated state/pdf $\omega/p\colon D \to [0,1]$ sends $x \in D$ to the fraction $\frac{\omega(x)\cdot p(x)}{\omega \mathrel{>\!\!=} p}$, when the validity $\omega \mathrel{>\!\!=} p$ is non-zero. A channel $D \to E$ is given by a parameterised pdf $c\colon D \times E \to \mathbb{R}_{\geq 0}$, with $\int_E c(x,y)\,\mathrm{d}y = 1$ for each $x \in D$. Then $(c \ll \omega)(y) = \int_D \omega(x) \cdot c(x,y)\,\mathrm{d}x$ and $(c \gg q)(x) = \int_E c(x,y) \cdot q(y)\,\mathrm{d}y$.

We sketch an example that combines discrete and continuous probability. The setting is given by the capture and recapture methodology to estimate the size of a population in ecology. Imagine we are looking at a pond and we wish to learn the number of fish. We catch twenty of them, mark them, and throw them back. Subsequently we catch another twenty, and find out that five of them are marked. What do we learn about the number of fish?

The number of fish in the pond must be at least 20. Let's assume the maximal number is 300. We thus take the interval $[20, 300] \subseteq \mathbb{R}$ as domain. We start from the uniform distribution since we don't assume any prior knowledge about the distribution of fish.

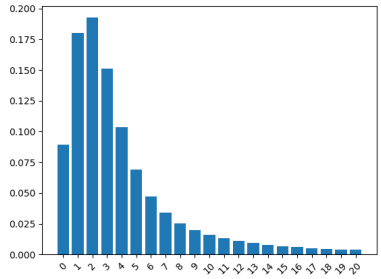
```
>>> fish_dom = R(20,300)
>>> prior = uniform_state(fish_dom)
>>> prior.expectation()
160.0
```

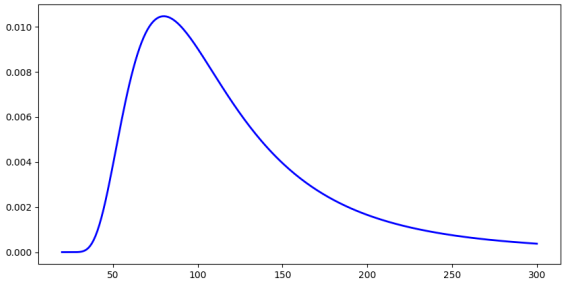
The method `expectation()` computes the mean value of the distribution.

We now assume that 20 of the fish in the pond are marked. We compute for each $x \in [20, 300]$ in the fish domain the probability of finding 5 marked fish when 20 of them are caught. For this we use the binomial distribution with parameters $N = 20$ and probability $p = \frac{x}{20}$. This is incorporated in the following channel, from the fish domain to the booleans.

```
>>> c = chan_fromklmap(lambda x: binomial(20, 20/x), fish_dom, range(21))
>>> (c >> prior).plot()
```

The latter plot command produces the (discrete) distribution on the left below. It gives the probability for each $k \in \{0, 1, \ldots, 20\}$ of catching k marked fish, in the prior uniform state.



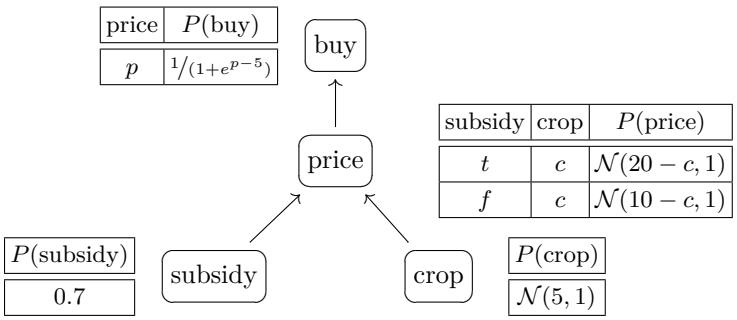


The plot on the right describes the posterior state that is obtained by updating the prior with the information that five marked fish have been found. The latter is expressed via a transformed predicate, as below.

```
>>> posterior = prior / (chan << point_pred(5, range(21)))
>>> posterior.plot()
>>> posterior.expectation()
116.491929836
```

The expected number of fish in the pond after recapture is calculated as 116.

We include another illustration, namely a *hybrid* Bayesian network, in which discrete and continuous probability are combined. Consider the following network from [2], with two discrete nodes (namely subsidy and buy) and two continuous ones (crop and price). The normal price distribution depends continuously on the crop, which is used as input to the its mean, in a way that depends whether subsidy is given. Whether or not the product will be bought is described as a biased coin, with bias parameter given by a 'sigmoid' function.



The conditional probability tables describe how nodes depend on each other. We write $\mathcal{N}(\mu, \sigma)$ for the normal (Gaussian) distribution with mean μ and standard deviation σ.

This network is modeled in EfProb in the following way. First, the subsidy and corp distributions form a joint prior state:

```
>>> subsidy = flip(0.7)
>>> dom = R(0,20)
>>> crop = gaussian_state(5,1,dom)
>>> prior = subsidy @ crop
```

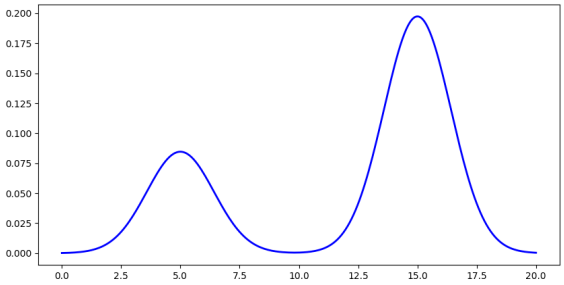
As indicated, the domain of the crop distribution is the real interval $[0, 20]$. The above two conditional probability for price and buy are translated into channels:

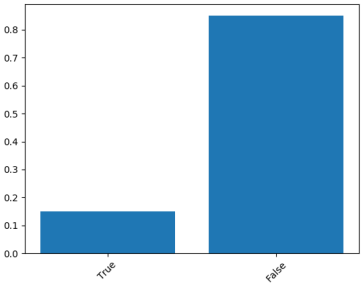
```
>>> price = chan_fromklmap(lambda p,c: gaussian_state(20-c, 1, dom) if p
...                        else gaussian_state(10-c, 1, dom),
...                        [bool_dom, dom], dom)
>>> buy = chan_fromklmap(lambda p: flip(1/(1+exp(p-5))), dom, bool_dom)
```

We see that the translation from the Bayesian network to the EfProb code is rather straightforward. We can now do 'forward' calculations to obtain distributions for price and buy by state transformation:

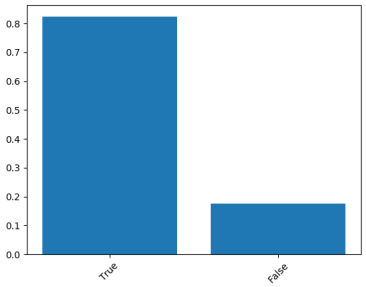
```
>>> p = price >> prior
>>> p.expectation(), p.variance()
11.9997758228 22.9911228888
>>> b = buy >> p
>>> b >= yes_pred
0.150080181436
```

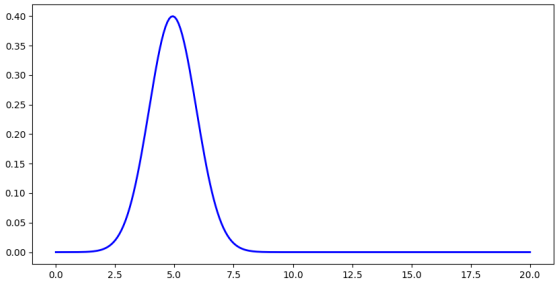
Computing these numbers takes a few seconds, on an ordinary laptop. These values are (very close to) the ones reported in [2]. The plots of the computed prior price and buy distributions (`p` and `b`) are given below. We see that the price distribution is a convex sum of two normal distributions.





We can also do 'backward' reasoning when we observe, for instance, that the product is not bought. The resulting posterior subsidy and crop distributions become:





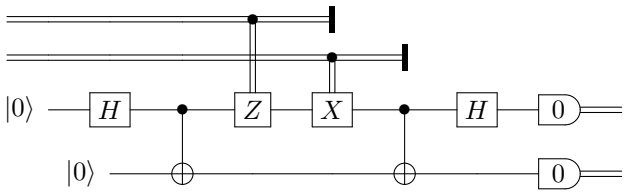
We see that the subsidy probability increases to 82%; the crop distribution looks unchanged, but its mean actually drops from 5 to 4.9. This update prior is produced via predicate transformation, using a sequential composition channel `buy ∗ price` in:

```
>>> posterior = prior / ((buy * price) << no_pred)
>>> (posterior % [1,0]).plot()
>>> (posterior % [0,1]).plot()
```

The two marginals of the posterior state, for subsidy and crop, are plotted.

5 Quantum probability

A quantum state of dimension n is an $n \times n$ complex matrix ω that is positive and has trace one: $\mathrm{tr}(\omega) = 1$. A predicate p is a positive matrix below the identity matrix: $0 \leq p \leq \mathrm{id}$. The validity $\omega \mathrel{>\!\!=} p$ is defined via the Born rule as $\mathrm{tr}(\omega\, p)$. The conditioned state ω/p is $\frac{\sqrt{p}\,\omega\sqrt{p}}{\omega \mathrel{>\!\!=} p}$. A channel $n \to m$ is a positive unitary linear map from $m \times m$ matrices to $n \times n$ matrices. We refer to [9] for more background information.



We briefly describe the so-called superdense coding protocol that is represented as a circuit above, using notation like in Quipper [4] and QWire [10]. In this protocol Alice transfers two classical bits to Bob via two entangled qubits — in the form of a Bell state that is produced in the lower-left part of the circuit — where Alice possesses one qubit and Bob the other. The EfProb code for Alice, Bob, and the whole protocol is described via sequential $*$ and parallel composition `@` of channels:

```
>>> alice = (discard(2) @ idn(2)) * ccontrol(x_chan) \
...          * (idn(2) @ discard(2) @ idn(2)) \
...          * (idn(2) @ ccontrol(z_chan)) * (swap @ idn(2))
>>> bob = (meas0 @ meas0) * (hadamard @ idn(2)) * cnot
>>> sdc = bob * (alice @ idn(2)) * (idn(2,2) @ bell00.as_chan())
```

We can run this protocol with two classical (probabilistic) states as input:

```
>>> s = random_probabilistic_state(2)
>>> t = random_probabilistic_state(2)
>>> s
[[ 0.89708576  0.        ]
 [ 0.          0.10291424]]
>>> (sdc >> s @ t) % [1,0]
[[ 0.89708576+0.j  0.00000000+0.j]
 [ 0.00000000+0.j  0.10291424+0.j]]
```

Similarly the second marginal of `sdc >> s @ t` yields the original second input `t`.

References

1. K. Cho, B. Jacobs, A. Westerbaan, and B. Westerbaan. An introduction to effectus theory. Preprint, 2015. arXiv:1512.05813 [cs.LO].
2. B. Cobb and P. Shenoy. Inference in hybrid Bayesian networks with mixtures of truncated exponentials. *Int. J. Approx. Reasoning*, 41(3):257–286, 2006.
3. N. Goodman, V. Mansinghka, D. Roy, K. Bonawitz, and J. Tenenbaum. Church: a language for generative models. In *Uncertainty in Artificial Intelligence*, 2008.
4. A. Green., P. LeF. Lumsdaine, N. Ross, P. Selinger, and B. Valiron. Quipper: A scalable quantum programming language. In *Programming Language Design and Implementation*, 2013.
5. B. Jacobs. New directions in categorical logic, for classical, probabilistic and quantum logic. *Logical Methods in Comp. Sci.*, 11(3):1–76, 2015. doi:10.2168/LMCS-11(3:24)2015.
6. B. Jacobs and K. Cho. EfProb user manual. 2017. URL: https://efprob.cs.ru.nl.
7. B. Jacobs and F. Zanasi. A predicate/state transformer semantics for Bayesian learning. In L. Birkedal, editor, *Math. Found. of Programming Semantics*, number 325 in Elect. Notes in Theor. Comp. Sci., pages 185–200. Elsevier, Amsterdam, 2016.
8. B. Milch, B. Marthi, S. Russell, D. Sontag, D. Ong, and A. Kolobov. BLOG: Probabilistic models with unknown objects. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
9. M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.
10. J. Paykin, R. Rand, and S. Zdancewic. QWIRE: A core language for quantum circuits. In *Princ. of Programming Languages*, pages 846–858. ACM Press, 2017.
11. F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, 2014.